195 PTAS.
(IVA Incluido)

112

MI COMPUTER

CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

P.V.P. Canarias, Ceuta y Melilla 185 Ptas.

VIDEOTEX ENABLES USERS TO ACCESS HIGH QUALITY, FULL-COLOUR OR BLACK AND WHITE PHOTOGRAPHS, PLUS TEXT AND GRAPHICS, FROM A CENTRAL DATA BASE.



MI COMPUTER CURSO PRACTICO
DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen X-Fascículo 112

Director:	José Mas Godayol
Director editorial:	Gerardo Romero
Jefe de redacción:	Pablo Parra
Coordinación editorial:	Jaime Mardones Francisco Martín
Asesor técnico:	Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Aribau, 185, 1.°, 08021 Barcelona
Tel. (93) 209 80 22 - Télex: 93392 EPPA

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 120 fascículos de aparición semanal, encuadernables en diez volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

© 1983 Orbis Publishing Ltd., London
© 1984 Editorial Delta, S. A., Barcelona

ISBN: 84-85822-83-8 (fascículo) 84-7598-183-6 (tomo 10)
84-85822-82-X (obra completa)
Depósito Legal: B. 52-84

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 118603

Impreso en España-Printed in Spain-Marzo 1986

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (120 fascículos más las tapas, guardas y transferibles para la confección de los 10 volúmenes) son las siguientes:

a) Un pago único anticipado de 27 105 ptas. o bien 10 pagos trimestrales anticipados y consecutivos de 2 711 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S. A. (Aribau, 185, 1.°, 08021 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

No se efectúan envíos contra reembolso.





Los sistemas de videotex envían imágenes de alta calidad a los ordenadores a través de las líneas telefónicas o las ondas



Crispin Thomas

Un universo ante sus ojos
La tecnología del videotex permite que los usuarios transmitan imágenes fotográficas, textos y gráficos por ordenador a terminales remotos, generando, en consecuencia, muchas aplicaciones diferentes en el hogar y en el comercio. Las noticias locales, la publicidad clasificada y el punto de información de ventas se pueden procesar y enlazar a sistemas interactivos para facilitar la realimentación del usuario. Por ejemplo, podrían llegar a ser de uso corriente gestiones tan disímiles como realizar las reservas para unas vacaciones en el extranjero y —en el futuro— comentar y contemplar las propuestas de planificación local

Transmisión visual

La mayoría de los usuarios de ordenadores personales se han acostumbrado a un nivel de definición en los gráficos muy inferior al de la calidad fotográfica. No obstante, un estándar de imagen más elevado es sólo una más de las características técnicas de lo que ha llegado a conocerse como *videotex*. Suscita confusión el hecho de que la palabra haya adquirido dos significados relacionados entre sí pero diferentes, según se la utilice en Gran Bretaña o en Estados Unidos. El primero cubre información visual y textual distribuida ya sea por un canal de televisión o por cable. En Estados Unidos, el término *videotex* se aplica a una combinación de texto y video transmitida exclusivamente por cable.

El inconveniente de todo sistema de videotex de gran calidad es la velocidad a la cual aparecen las imágenes en la pantalla. Mientras que una página de videotex típica del servicio Prestel de la British Telecom suele demorar menos de un segundo en llenar la pantalla, en otros sistemas puede tardar 10 segundos o más. La razón estriba en la cantidad de datos que es necesario transmitir para que aparezca una única pantalla. Sería muy ilustrativo comparar los sistemas Prestel y Photo Videotex de la British Telecom.

Las definiciones de pantalla se cuantifican en pixels. Una página de Prestel suele ser de 234 pixels por 22 líneas. Por el contrario, una pantalla de Photo Videotex tiene una definición de 270 pixels por 240 líneas, pero en contrapartida por esta mayor definición es preciso satisfacer unas exigencias de datos muchísimo mayores para almacenar tal imagen. (El Photo Videotex requiere 128 Kbytes, mientras que una página de Prestel necesita 5 Kbytes como máximo.)

La compresión de datos está jugando un papel cada vez más importante en la tarea de rebajar las exigencias de almacenamiento de datos. El Photo Videotex, por ejemplo, es capaz de comprimir unas exigencias teóricas de 128 Kbytes en 64 Kbytes reales. Con el tiempo, los sistemas "inteligentes" podrán muestrear la pantalla y enviar datos sólo a las partes de la imagen que hayan cambiado. Así se reducirá drásticamente la cantidad resultante de datos que sea necesario enviar.

El Videotex se puede utilizar para cualquier aplicación en la que se necesite combinar texto con ilustraciones en color. Por ejemplo, en los sistemas de seguridad, que son los principales usuarios, el videotex ayuda a emparejar rostros con nombres. Con esta disposición, un guarda puede consultar los registros de monitor retenidos en una base de datos central. Al digitar un nombre, aparecerá un registro conteniendo una foto en color de la persona más cierta cantidad de información de carácter personal. En el caso de que el guarda no estuviera seguro de que la persona corresponda a la fotografía, es una tarea sencilla efectuar una doble comprobación y pedir a la persona en cuestión la fecha de nacimiento, el número del documento nacional de identidad, etc.

Asimismo, el videotex es un importante medio publicitario. Los agentes de viajes y los agentes inmobiliarios ya se valen de sistemas de videotex para poner en contacto a compradores y vendedores. Futuros desarrollos de este enfoque podrían conducir a que se colocaran pantallas en los escaparates de las tiendas y se los hiciera pasar por fotogramas retenidos en una base de datos. Los viandantes, por lo tanto, podrían ver detalles de las últimas ofertas o de las fincas que se encuentran en venta. Puesto que el videotex es interactivo, también se podrían transmitir las decisiones de compra. Los agentes de viajes, por ejemplo, podrían realizar reservas con las agencias. Si se estuviera utilizando una línea telefónica o una red privada, las dos firmas no necesitarían siquiera hallarse en el mismo país.

En Gran Bretaña, por intermedio de un servicio independiente llamado Armchair Grocer (tendero de sillón) y a través de un cable de televisión, se

pueden hacer compras con el Prestel (Club 403). El videotex, incluso, permite a los usuarios curiosear en los catálogos comerciales en la intimidad de sus hogares. Las compras se pueden efectuar dando un número de tarjeta de crédito o bien mediante transferencia de fondos electrónica directa.

Las fronteras del videotex ya se están ensanchando para incluir las "videoconferencias". Una empresa norteamericana, Datapoint, ha producido un dispositivo denominado Minx, que contiene una cámara de video, un teléfono con altavoz, una pantalla en color y otra de referencia mucho más pequeña. Utilizando una red de área local (Arcnet) o línea de datos, las dos partes pueden llamarse y ver a la persona con quien están hablando. El Minx posee facilidades añadidas, puesto que también es compatible con el IBM PC. La línea que está portando la llamada de voz tomará simultáneamente las señales de video y de datos. En consecuencia, ambas partes pueden ver visualizados los mismos datos durante la conversación, contemplándose aun a sí mismos en las pantallas de referencia más pequeñas. De la misma manera se pueden intercambiar archivos. Tal sistema exige un enlace de datos de dos megabytes por segundo.

Inicialmente, los servicios de videotex requerían un ordenador central o un miniordenador para almacenar los datos y gestionar las llamadas. Esta situación está experimentando un acelerado cambio gracias a los micros. Por ejemplo, a pequeña escala, en Gran Bretaña —país pionero en el campo de la informática—, la firma Soft Machinery está utilizando el BBC Micro para operar un servicio denominado The Gnome at Home (El gnomo en casa). En efecto, al unir varios micros entre sí mediante una red Econet, se hace posible que hasta ocho usuarios llamen simultáneamente. El software es una versión modificada de Communitel, un paquete desarrollado por Notting Dale ITEC como anfitrión de videotex para el BBC Micro. En muchos sentidos, Communitel es similar a un tablón de anuncios, si bien es compatible con el Prestel.

Apricot ha producido el Apricot Viewdata, que proporciona un sistema anfitrión de videotex privado para entre 8 y 16 usuarios simultáneos. Utilizando puertas RS232, se accede a la opción a través de un conjunto de modems o una red de área local. Al ser compatible con Prestel, se lo puede conectar al sistema Prestel propiamente dicho a través de un *gateway*. (*Gateway* es el término genérico para designar un enlace entre dos ordenadores en un sistema de videotex.)

El sistema Photo Videotex de British Telecom también utiliza un microordenador (el IBM PC) que combina una entrada de imágenes en color o en blanco y negro proveniente de una cámara PAL o RGB con texto estándar. La calidad de la imagen es comparable a la de una fotografía en color.

El videotex puede aportar soluciones a una gran variedad de problemas. En la medida en que se vaya extendiendo el uso de las redes de área local y los enlaces telefónicos digitales —por fibra óptica y por satélite—, los terminales de videotex se convertirán en una presencia familiar en la industria, en el comercio e incluso en el hogar.

Cortesía de British Telecom

Videotex y teletexto

Para distinguir los dos tipos de videotex que por lo general se utilizan en Gran Bretaña se han acuñado los términos *videotex* y *teletexto*. El videotex se basa en el trabajo que realizó Sam Fedida en lo que se ha convertido en el Centro de Investigación de British Telecom en Martlesham Heath. Como resultado de sus propuestas, en 1979 nació el primer sistema de videotex del mundo disponible comercialmente, en la forma del Prestel. La forma de videotex del Prestel ha obtenido un gran éxito comercial, permitiéndole penetrar en un mercado tan marcadamente competitivo como el de Estados Unidos. Sin embargo, el Prestel no es de ningún modo el único servicio de videotex en Gran Bretaña. Existen servicios de videotex privados; ADP e Istel (subsidiaria de British Leyland) poseen sus propios sistemas accesibles por teléfono.

Si bien el videotex se basa fundamentalmente en las líneas telefónicas, las transmisiones de teletexto se adaptan muy bien a la señal de televisión estándar. Nuevamente, el resultado es una combinación de texto y gráficos. Para poder



Cortesía de MTV

Imágenes cableadas

Si bien la televisión por cable es un medio ideal para las transmisiones de videotex, en Gran Bretaña es un competidor de poca altura. Los informes de 1985 señalaban que sólo 127 000 familias recibían televisión por cable. Cuando las jóvenes empresas de cable como Westminster Cable estén totalmente en actividad, esta cifra se aproximará a 300 000. En 1984 se llevó a cabo un experimento en transmisión de software por cable. Llamado Proyecto Gamestar, fue una empresa conjunta entre British Telecom y Micronet 800. Los usuarios pudieron alquilar un Spectrum y un modem Prism modificado para cargar desde una línea software de juegos. Fue secundado por algunas empresas de cable incluyendo a Rediffusion, pero el proyecto fue abandonado debido a dificultades técnicas. Hoy British Telecom se está concentrando en el Cabletext, un derivado de Prestel para empresas de cable



Pase privado
El sistema Photo Videotex de la British Telecom utiliza un IBM PC y proporciona amplias facilidades de edición, permitiendo cortar las imágenes, desplazarlas y aplicarles *zoom*, y combinarlas con superposiciones, texto y gráficos mejorados. Entre las aplicaciones se incluyen noticias locales, publicidad, promociones de puntos de venta, registros de personal y sistemas caseros para grandes empresas. El sistema enlaza terminales de usuarios, terminales de edición y un sistema de ordenador central, permitiendo la comunicación en dos sentidos y una respuesta directa. La calidad de la imagen es comparable a la de un televisor doméstico y cada imagen en color de tamaño completo requiere 64 Kbytes de almacenamiento

tish Telecom 388414a 0p
VISIT
BANGKOK
WITH
TRANSWORLD'S
SOUTH EAST
ASIA EXPRESS
TICKET
PHONE
01 733 6210
FOR DETAILS
R PICTURE 0 TRANSWORLD INDEX

recibir teletexto se necesitan aparatos de televisión modificados especialmente. El teletexto emplea páginas tal como su equivalente de videotex, pero, a diferencia de éste, no es del todo interactivo y no puede transmitir las respuestas del usuario.

En algunas ocasiones se alude al teletexto como "videotex emitido" (no se debe confundir el término con "teletex", que es una forma mejorada de telex). La cooperación técnica entre la British Telecom y la BBC/IBA ha dado lugar a un grado de compatibilidad entre los dos sistemas. En consecuencia, se ha aplicado el término "*world system*" (sistema mundial) a todo cuanto se ajuste a los estándares británicos Prestel u Oracle/Ceefax.

Tradicionalmente, los fabricantes norteamericanos han preferido un grado de resolución de imagen mayor que el empleado por el Prestel. Es así como sus sistemas de videotex mezclan imágenes de video generadas por una cámara con el texto de un ordenador. Norteamérica no cuenta con un equivalente de teletexto. Por razones de claridad, aquí se alude a los sistemas que utilizan televisión como *teletexto*, mientras que *videotex* se empleará para cualquier tipo de sistema que emplee cable o líneas telefónicas

El videotex británico

El tipo de cable utilizado por un sistema de videotex puede ser de muy variadas formas. Un sistema de videotex podría formar parte de una red de área local, por ejemplo. El cable que realiza la conexión física puede ser un par arrollado de hilos (apenas dos cables), un cable coaxial (como una antena de televisión) o un cable de varios hilos (como en el cable RS232 utilizado para impresoras y modems).

Alternativamente, se puede utilizar a modo de cable una línea de la British Telecom. La línea telefónica normal que se utiliza para una llamada de voz cotidiana forma parte de la Public Switched Telephone Network. Una línea PSTN puede unir el extremo receptor directamente con la fuente de videotex, como en el caso del servicio Prestel de la British Telecom.

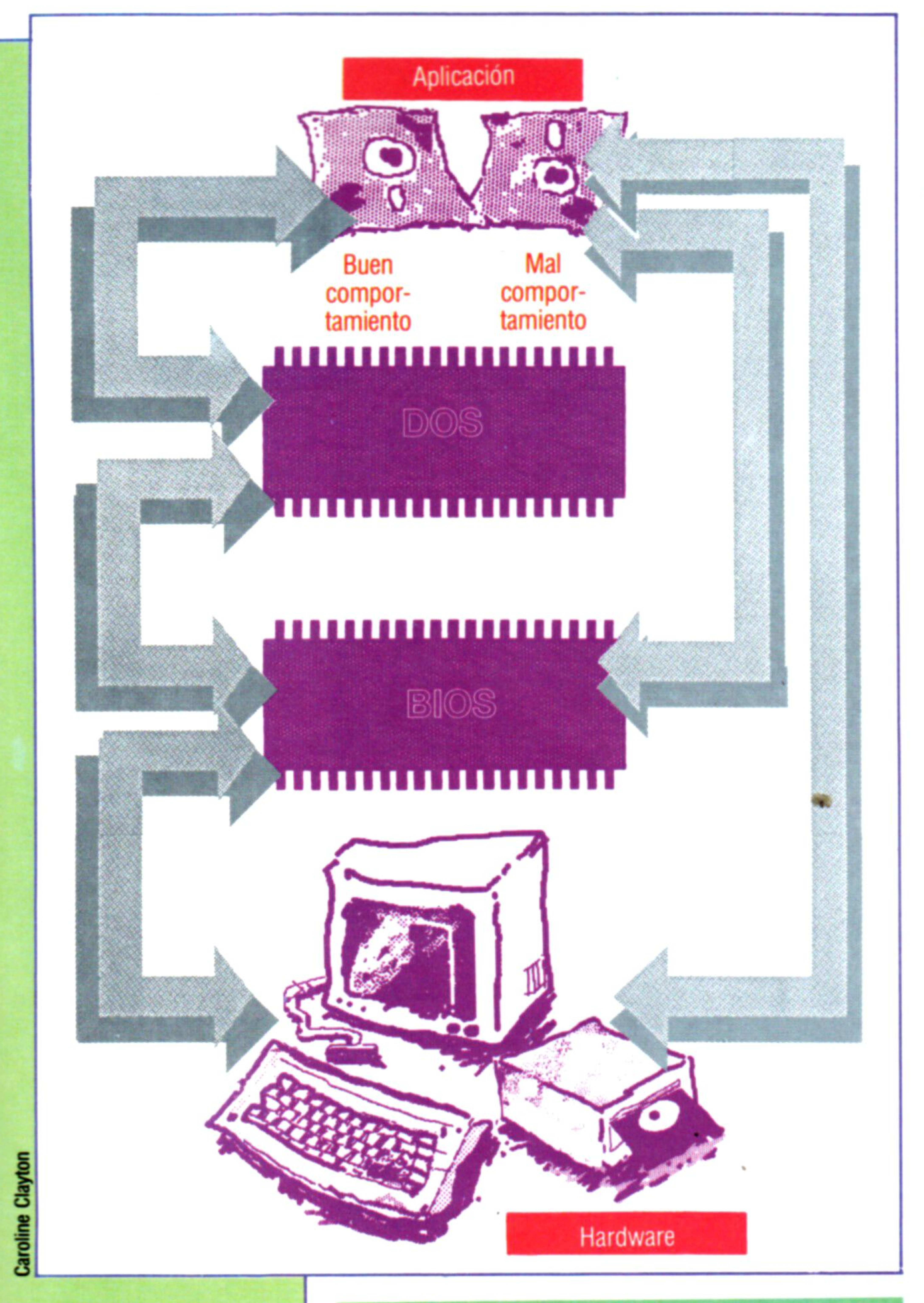
El usuario del micro puede acceder al Packet SwitchStream (PSS) a través de la PSTN. Utilizando un modem se efectúa una llamada local a lo que se conoce como un "nudo". El nudo toma los datos y los convierte en *packets* (paquetes; en realidad, lotes de datos autocontenidos) bajo un protocolo denominado X25. El nudo posee sus propios enlaces de datos directos con otros nudos diseminados a través del país. El servicio cruza incluso las fronteras y está enlazado con servicios similares de al menos otros 20 países. El servicio se denomina International Packet SwitchStream (IPSS). La conmutación de paquetes puede hacer frente a velocidades de transmisión de datos de hasta 48 Kbits por segundo, y es tan bueno como el enlace con el nudo local: si la línea es ruidosa, se necesita comprobación de errores y disminuyen las velocidades de transmisión.

Por este motivo es preferible utilizar una línea privada, exclusiva, que transmitirá los datos con mucha más limpieza. Las líneas contratadas privadas conforman los servicios "*X-stream*" de la British Telecom, tales como MultiStream, KiloStream, MegaStream e incluso SatStream (un enlace por satélite). El rival de British Telecom, Mercury, también puede proporcionar líneas de datos exclusivas.

Tales servicios son en realidad medidas provisionales hasta que la totalidad del país se convierta en la red de intercambio telefónico digital de la British Telecom, el System X, para finales de esta década. La capacidad del System X para conmutar datos además de llamadas de voz hará del videotex una fácil opción.

En 1985, un sistema de videotex que produjera un enlace de calidad fotográfica requería una velocidad de transmisión de datos de entre 64 Kbits y dos Mbytes por segundo

Portador estándar



Buena conducta

Se dice que el software que hace uso de llamadas MS-DOS estándares es de "buen comportamiento". En este caso, el DOS llama a la rutina pertinente BIOS indirectamente a través de una "tabla de salto". En el diagrama se ilustra esta ruta de "buen comportamiento". Se considera que los programas que presuponen el conocimiento de direcciones absolutas en BIOS, o que se saltan al MS-DOS completamente al "hablar" directamente al hardware subyacente, tienen un "mal comportamiento". En el IBM PC en particular, el BIOS es un chip de ROM patentado y la tabla de salto comienza en la dirección 400H. Ésta está situada 512 bytes (dos páginas hexadecimales) por debajo de la base de tablas MS-DOS estándar, en 600H. Por consiguiente, un programa que llame al BIOS o al hardware directamente, sólo se ejecutará en una máquina IBM o un clono 100% compatible

Luego de estudiar el CP/M de Digital Research, estándar de la industria de 8 bits, examinemos el MS-DOS de Microsoft, que ha surgido como el estándar de 16 bits

En los primeros días de los microordenadores de ocho bits, la industria estaba dominada por máquinas que utilizaban el procesador Zilog Z80 y CP/M. Este sistema operativo de disco lo había escrito originalmente Gary Kildall para el chip Intel 8080; posteriormente Kildall crearía Digital Research.

Para cuando IBM decidió entrar en el mercado del micro, los sistemas de ocho bits se estaban acercando al fin de su vida útil; pero aún no había surgido una combinación definitiva de CPU y sistema operativo como alternativa de 16 bits para el Z80 y el CP/M. Por consiguiente, IBM tuvo la opción de introducir una configuración de hardware y software completamente nueva para el mercado de 16 bits. Decidieron no hacerlo por diversos motivos:

1. A pesar de dominar el mundo de los ordenadores centrales, IBM no tenía experiencia en el mercado de los micros, que era radicalmente diferente.
2. El éxito de DEC (Digital Equipment Corporation) en el campo del miniordenador le había demostrado a IBM que los usuarios ya no estaban predispuestos a adquirir productos de IBM sin una evaluación seria del producto.
3. La cantidad de tiempo requerida para desarrollar una base de software adecuada para cualquier sistema nuevo hubiera sido inaceptable en el mundo del micro, en el cual los cambios se suceden con gran rapidez, a menos que IBM obtuviera el apoyo de casas de software independientes.
4. Los costos de desarrollo tanto del software como del hardware habrían significado unos precios de venta al público inaceptablemente elevados.

En consecuencia, se tomó la decisión de adoptar lo "mejor" de lo que hubiera disponible entonces (es decir, todo lo que predominara en el mercado) y adaptarlo en la medida de lo necesario para sistemas de 16 bits.

Este enfoque tan poco innovador dio como resultado el IBM PC, máquina conservadora en cuanto a diseño, lenta en ejecución y que no abre ninguna brecha, especialmente si se la compara con sus contemporáneas, el Apple Lisa, por ejemplo. Incluso la elección del procesador Intel 8088 y un compromiso de futuros desarrollos centrados en las series de chips 8086/186/286/386 (conocidos generalmente como la familia 8086) fue una decisión pragmática. Si bien el 8088 tenía una arquitectura interna de 16 bits, el bus de datos externo era sólo de ocho bits de anchura. Esto supone que cada "palabra" de 16 bits se busca y trae en dos operaciones, reduciendo, por tanto, la velocidad de proceso, pero permitiendo el uso de chips periféricos de ocho bits existentes ya probados (y baratos).

Para el sistema operativo y el software de aplicaciones IBM acudió a Microsoft, empresa que había ganado prestigio en la microinformática fundamentalmente por su difundida versión de BASIC. En aquel entonces, Microsoft estaba en contacto con una firma denominada Seattle Computer Products, que estaba trabajando con sistemas de 16 bits basados en el Intel. En ausencia de un CP/M para la familia 8086, un programador llamado Tim Patterson había escrito un DOS en lenguaje máquina (denominado SCP-DOS) que había modelado siguiendo muy de cerca el CP/M. Su sistema también se conoció como QDOS (no confundir con el OS de Sinclair para el QL), y estas siglas respondían a la frase "*quick and dirty operating system*" (sistema operativo rápido y sucio). Esa descripción se le adecuaba a la perfección; el OS de Patterson era utilizable, pero no estaba totalmente desarrollado

ni tampoco depurado. Microsoft adquirió los derechos para el sistema y les concedió licencia de uso a los fabricantes de equipos originales (*original equipment manufacturers*: OEMs).

PC-DOS

IBM produjo su propia versión (versión 1) del MS-DOS para el IBM PC, denominada PC-DOS, y desde entonces ha estado al día con los principales desarrollos en ulteriores versiones del MS-DOS; a partir de este momento, siempre que en esta serie mencionemos una versión MS-DOS, la referencia será igualmente aplicable al PC-DOS. Pero veamos primero algunas de las diferencias entre ambos. Por cuanto concierne al usuario, esencialmente no existe ninguna. Algunos detalles internos difieren, y varían algunos detalles irrelevantes tales como la numeración de las unidades de disco; pero eso es todo, al menos en teoría. Lamentablemente, sin embargo, se han escrito muchos programas de aplicaciones para el IBM PC que direccionan el hardware directamente, en lugar de utilizar las llamadas al sistema operativo proporcionadas. Fundamentalmente por este motivo surgen problemas de compatibilidad (no es un problema inherente a las diferentes versiones de MS-DOS y PC-DOS). A los programas que llaman al DOS correctamente se los suele describir como "de buen comportamiento", y tales programas se suelen ejecutar sin modificación alguna en cualquier máquina que posea ya sea MS-DOS o bien PC-DOS de la misma versión o de versiones *recientes*. Naturalmente, no se puede esperar que las operaciones que hacen uso de las flamantes mejoras al DOS en, por ejemplo, la versión 3.1, funcionen a la perfección cuando se las ejecute en una versión anterior que no soporte esas determinadas facilidades.

La primera versión del MS-DOS era esencialmente una versión de CP/M recodificada para la familia de 16 bits de Intel. Si usted se remite a nuestra serie dedicada al CP/M, reconocerá muchas de las instrucciones y observará las semejanzas de estructura interna. No obstante, Tim Patterson sí cambió varios detalles y facilitó la labor del usuario en muchas instrucciones.

Cuando Microsoft comenzó a trabajar en las mejoras, tuvo en mente el futuro uso del Unix (el sistema operativo multiusuario de los Laboratorios Bell) y, en particular, la versión Microsoft, Xenix. Se pretendía un cierto grado de compatibilidad entre los dos, con la implementación de llamadas comunes al sistema. Pero, con mucho, el adelanto más importante en DOS 2 desde el punto de vista del usuario fue la introducción de la "redirección" de E/S, *pipelines* y "directorios jerárquicos", todos los cuales fueron ideas tomadas del Unix. En realidad, podemos definir al DOS 2 como "el DOS 1 más un poco de Unix".

DOS 3

La versión más reciente de MS-DOS (DOS 3) incorpora todas las facilidades del DOS 2 con la adición de soporte multiusuario. De las dos versiones subsidiarias principales (3.0 y 3.1), los OEMs que proporcionan sistemas para conexión en red, como Apricot y Research Machines, están utilizando la versión 3.1 con Microsoft Networks incorporado.

Clónico

Los primeros problemas de compatibilidad surgieron al desplazar IBM una "tabla de salto" dos páginas (hexadecimales) hacia abajo en la memoria y colocarla en una ROM en 400H (la tabla MS-DOS empieza en 600H). Esta parte del BIOS (sistema básico de entrada/salida) está sujeta al *copyright* de IBM, y plantea problemas para otros fabricantes que desean construir máquinas 100% compatibles con IBM ("clonos"). Sin embargo, si todo el acceso al hardware subyacente se realiza a través del MS-DOS por medio de las llamadas al sistema correctas, no existe ningún problema para producir software que se pueda ejecutar en cualquier ordenador MS-DOS.

Hay, no obstante, un pequeño precio a pagar por ello. Las llamadas al sistema le añaden una ligera penalización en tiempo a la velocidad de ejecución y, lo que quizá resulte más significativo, puede ser que no se comprendan las ventajas resultantes de unas especificaciones de hardware más avanzadas. Por estos motivos, varias casas de software han escrito programas dependientes de hardware (o firmware) que necesitan un entorno de hardware que se corresponda más íntimamente con el del IBM PC. Tampoco es un hecho desconocido el que los vendedores de software hagan deliberadamente que sus productos no sean portables. Digital Research, por ejemplo, altera unos pocos bytes de código para hacer que sea necesario comprar versiones diferentes (de aplicaciones GEM, p. ej.) para cada máquina, aun para el más compatible de los clonos

Esto, sin embargo, puede hacer que el sistema operativo sea innecesariamente grande, de modo que se dispone de la DOS 3.0 para otros entornos, privada de la capacidad para conexión en red de la 3.1, pero conservando las facilidades para compartir archivos y proteger registros. La solución por software proporciona una respuesta práctica y eficaz a los problemas de ejecutar aplicaciones que necesitan permitir el acceso a bases de datos comunes.

El último procesador de Intel (80286), utilizado en el IBM PC-AT, posee tanto una capacidad de direccionamiento de memoria de tres Mbytes como la facilidad, incorporada en el hardware, de proteger regiones de archivos. El DOS 3.0 permite compartir archivos en el 8086 y el 80186 mediante llamadas adicionales al sistema que pueden prohibir temporalmente el acceso a una región. Esto impide la "colisión de acceso", pero a expensas de la velocidad de proceso (por el tiempo extra que requieren las llamadas al sistema). Se han introducido otras mejoras y, si bien aún pueden hallarse uno o dos vestigios del DOS "rápido y sucio" de Seattle, el MS-DOS se puede considerar como un producto depurado con un futuro por delante.

Entre las mejoras más importantes que hay disponibles en la actualidad está Microsoft Windows. Al igual que el GEM de DR, proporciona un entorno tipo WIMP junto con facilidades para multitareas y operaciones de "cortar y pegar" entre aplicaciones. Se está implementando como una ampliación al sistema operativo MS-DOS/PC-DOS básico, y está diseñado para proclamar una nueva era en la sencillez de uso de ordenadores convencionales.


Chris Stevens

Ostentoso dominio
El dominio del IBM PC y su adopción del MS-DOS (bajo el nombre de PC-DOS) le ha asegurado al MS-DOS el primer puesto entre los OS de gestión de 16 bits. El sistema nació como una versión de CP/M hecha a medida, rudimentaria pero efectiva, por un ex empleado de Digital Research. Luego fue adquirida por Microsoft, que le dio la licencia a IBM

Un juego en regla

Nos ocuparemos de las rutinas encargadas del almacenamiento y de la evaluación de cada mano

Esbozaremos las reglas por las que se regirá nuestro programa; existen tantas versiones del juego como nombres del mismo. No obstante, todas tienen varios aspectos fundamentales en común, siendo el más importante el que el jugador intenta acercarse lo más posible a la puntuación de 21 sin pasarse. Las cartas de nuestro programa llevan los siguientes valores: los naipes numerados valen en función de los valores de sus caras, los naipes de figura (Valet, Dama y Rey) valen 10, y los ases valen uno u 11, a discreción del jugador.

En su forma más simple, en el veintiuno juega un jugador contra la "banca", que reparte dos naipes de la parte superior de la baraja: uno boca arriba para el jugador y otro boca abajo para sí misma. Luego se reparte otro naipe cara arriba a cada uno, en cuyo momento el jugador debe decidir si "plantarse", si su mano está en 21 o cerca de 21, o "doblar" y recibir otro naipe. El riesgo, por supuesto, es que el jugador pueda "pasarse" de 21. Después de que el jugador se plante, la banca jugará su mano hasta vencer al jugador o bien pasarse.

Dándonos una mano
Se utiliza una matriz tridimensional, HD(,,), para retener las manos del jugador y la banca a medida que se reparten los naipes de la baraja. Una segunda matriz, HP(), retiene punteros al siguiente espacio libre de cada sección de la matriz de manos. Al final de cada juego el programa no necesita limpiar la matriz; tan sólo debe restablecer los punteros a uno

Hemos introducido en este programa algunas reglas adicionales que se utilizan comúnmente en el veintiuno. En primer lugar, usted no puede plantarse si su mano es inferior a 17. En segundo lugar, si tras habérsele repartido los dos primeros naipes su marcador es de 12, 13 o 14, tiene la opción de "quemar" sus naipes, y hacer que se le repartan otros dos. En caso de empate entre usted y la banca, ésta siempre gana.

Aunque hemos visto cómo se reparten los naipes de la baraja y cómo se visualizan, no le hemos mostrado cómo el programa podría "recordar" las cartas repartidas. Para retener las manos del jugador y de la banca, prepararemos una matriz tridimensional HD(,,). Dado que ninguno de los bandos requerirá nunca más de cinco naipes, esta matriz se dimensiona en la línea 550 para ser de dos elementos (para los dos jugadores) por cinco (para los naipes) por dos (para retener los números y palos por separado). Se utiliza una segunda matriz, HP(), para retener un puntero hacia el siguiente espacio libre en la matriz de manos para cada jugador.

En esta etapa necesitamos añadir la línea 1325 a la rutina para repartir desarrollada previamente, de modo que el número de naipe, CN, y el número de palo, SU, se añadan en la matriz de manos y se incremente el puntero de mano. Observe que si PL=1 al entrar esta rutina, el naipe repartido se entra en la mitad de la matriz de manos perteneciente al jugador; si PL=2, los detalles del naipe se entran en la mitad de la banca.

Ahora se pueden repartir los cuatro primeros naipes del juego estableciendo PL en 1 o 2 y llamando a la rutina para repartir. Puesto que la primera carta a repartir a la banca ha de estar boca abajo, la rutina para repartir está escrita de modo tal que si una bandera (FL) está en 1, se llama a la rutina para visualizar el reverso del naipe. Los detalles de la carta oculta se entran en la matriz de manos.

La subrutina de la línea 800 está diseñada como una rutina de evaluación con fines generales que cumple dos funciones. Primero, calcula el valor total retenido en la mano que se está examinando, y en segundo lugar establece la variable EF en un valor entre 1 y 5, correspondiente a los cinco estados posibles en los que puede estar una mano. El cálculo del estado de una mano es relativamente directo, pero lo que es más difícil es calcular el valor total de ésta. Si bien a primera vista parece que todo cuanto necesitamos hacer es totalizar los números de naipes retenidos en la matriz de manos, los naipes de figura valen 10 y los ases valen poco o mucho. El primer problema se puede resolver fácilmente comprobando el número de naipe; si éste es superior a 10, entonces se le sumará 10 al valor total.

El problema de los ases, sin embargo, requiere un poco más de atención, dado que una mano puede contener hasta cuatro ases, creando hasta 16 posibles permutaciones de valor con las que tratar. En realidad, no vale la pena considerar la mayoría de estas permutaciones, porque hacen que la mano se pase inmediatamente. En consecuencia, podemos seguir una sencilla regla para manejar los ases: el primer as de la mano puede valer uno u 11, pero todos los ases siguientes se cuentan en su valor bajo o la mano se pasará.

Para codificar esta regla, la forma más simple es tener dos opciones de valor: TT(PL,1), en la cual el primer as se cuenta bajo, y TT(PL,2), en la cual el primer as se cuenta alto. Utilizamos un único bucle para sumar los valores de las manos, tratando a todos los ases encontrados como bajos, y contando la cantidad de ases. Al salir del bucle, comprobamos la cuenta de ases y, si no es cero, le sumamos 10 a TT(PL,2). El resto de la rutina de evaluación comprueba los totales de TT(PL,1) y TT(PL,2) o la matriz de manos para determinar en qué estado se halla la mano y establece la bandera de estado de evaluación, EF, en consecuencia.

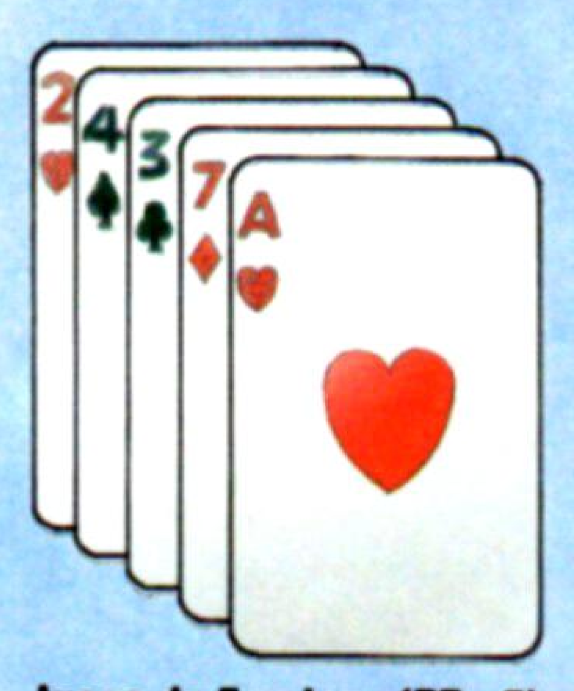
Si a usted se le han repartido dos naipes que totalizan 12, 13 o 14, tiene la opción de quemarlos y se le repartirán otros dos. Esta rutina llama a la rutina de evaluación analizada anteriormente y utiliza los totales de TT(PL,1) y TT(PL,2) para determinar si usted puede quemarlos. Si opta por hacerlo así, se borra la mitad izquierda de la visualización en pantalla, se restablece su puntero de mano, HP(PL), y se le reparten dos nuevos naipes.

Esquemas de valores

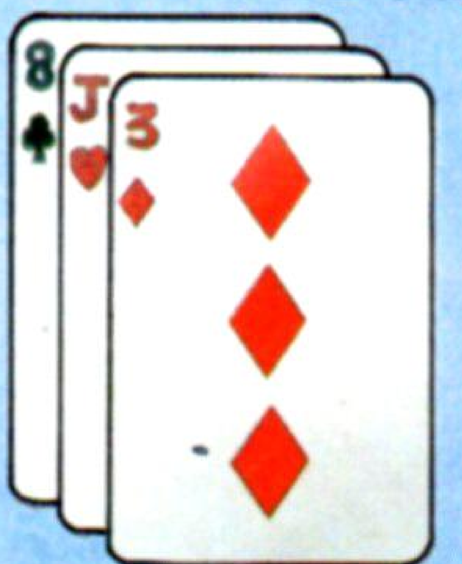


En el veintiuno *(pontoon)*, una mano puede estar durante el juego en cinco estados diferentes. La siguiente lista muestra los diversos estados que reconoce nuestro programa por orden jerárquico. La rutina de evaluación establece una variable, EF, según el estado en el que esté la mano

Royal pontoon (EF=2)
Un marcador de 21 compuesto por un as y una figura. Un as y un 10 no se consideran *royal pontoon*



Juego de 5 naipes (EF=5)
Un marcador de 21 o menos con cinco naipes



Pontoon (EF=3)
Un marcador de exactamente 21



Menos de 21 (EF=1)
Un marcador de menos de 21. Éste puede ser el estado de una mano al final de un turno o en alguna etapa intermedia



Pasarse, o *bust* (EF=4)
Un marcador de más de 21

Kevin Jones

Rutinas de almacenamiento y evaluación de mano

Sinclair Spectrum

```
60 > LET FL=0: LET PL=1: GO SUB 1300: REM REPARTIR
     NAIPE A APOSTADOR
70 LET FL=1: LET PL=2: GO SUB 1300: REM REPARTIR
     NAIPE A BANCA
85 LET FL=0: LET PL=1: GO SUB 1300: REM REPARTIR
     NAIPE A APOSTADOR
90 LET FL=0: LET PL=2: GO SUB 1300: REM REPARTIR
     NAIPE A BANCA
95 REM **** TURNO DEL APOSTADOR ****
100 LET PL=1
102 GO SUB 2300: REM OPCION QUEMAR
```

Dimensionar matrices de manos

```
550 > DIM D(2,5,2):DIM P(2): REM MANOS DEL APOSTADOR
      Y DE LA BANCA
555 DIM T(2,2): REM TOTALES PUNTUACION
670 REM **** BORRAR VISUALIZACION NAIPE ****
675 LET X(PL)=EP: LET Y(PL)=0
680 PRINT AT 0,0;: FOR I=1 TO 12: PRINT AT I,EP,S$(TO 7):
    NEXT I: RETURN
700 REM **** PREPARAR PARA ENTRADA ****
710 LET TX=0: LET TY=17: GO SUB 900: PRINT S$( TO 31)
720 LET TX=0: LET TY=17: GO SUB 900: RETURN
```

Rutina de evaluación de mano

```
800 REM **** EVALUAR MANO ****
810 LET AV=1: FOR J=1 TO 2
812 LET T(PL,J)=0
815 FOR I=1 TO P(PL)-1
820 IF D(PL,I,1)=1 THEN LET T(PL,J)=T(PL,J)+AV-1
825 IF D(PL,I,1)>10 THEN LET
    T(PL,J)=T(PL,J)+10-D(PL,I,1)
830 LET T(PL,J)=T(PL,J)+ D(PL,I,1)
840 NEXT I: LET AV=11: NEXT J
852 IF (T(PL,1)<=21) OR T(PL,2)<=21) AND P(PL)> 5
    THEN LET EF=5: RETURN
854 IF D(PL,1,1)=1 AND D(PL,2,1)> 10 THEN LET EF=2:
    RETURN
855 IF D(PL,2,1)=1 AND D(PL,1,1)> 10 THEN LET EF=2:
    RETURN
856 IF T(PL,1)=21 OR T(PL,2)=21 THEN LET EF=3: RETURN
858 IF T(PL,1)<21 OR T(PL,2)<21 THEN LET EF=1: RETURN
860 IF T(PL,1)> 21 AND T(PL,2)>21 THEN LET EF=4:
    RETURN
2300 > REM **** OPCION QUEMAR ****
2305 GO SUB 800: REM EVALUAR
2310 IF T(PL,1)<>T(PL,2) OR T(PL,1)<12 OR T(PL,1)> 14
     THEN RETURN
2340 GO SUB 700: PRINT "QUEMAR (S/N) ";
2345 LET A$=INKEY$: IF A$="" THEN GO TO 2345
2347 IF A$<> CHR$ 13 THEN PRINT A$
2350 IF A$<> "S" THEN RETURN
2360 LET P(1)=1: REM RESTABLECER PUNTERO MANO
2370 LET EP=0: GO SUB 670: REM BORRAR NAIPES
2380 LET FL=0: LET PL=1: GO SUB 1300: GO SUB 800: REM
     REPARTIR NAIPE A APOSTADOR
2390 LET FL=0: LET PL=1: GO SUB 1300: GO SUB 800: REM
     REPARTIR NAIPE A APOSTADOR
2400 GO TO 2300: REM QUEMAR OTRA VEZ?
```

BBC Micro

```
60 FL=0: PL=1: GOSUB 1300
70 FL=1: PL=2: GOSUB 1300
85 FL=0: PL=1: GOSUB 1300
90 FL=0: PL=2: GOSUB 1300
95 REM
100 PL=1
102 GOSUB 2300
```

Dimensionar matrices de manos

```
550 DIM HD(2,5,2),HP(2)
555 DIM TT(2,2)
670 REM **** BORRAR VISUALIZACION NAIPE ****
675 X(PL)=EP:Y(PL)=0
```

```
680 PRINT TAB(0,0);:FOR I=1 TO 12:PRINT
    TAB(EP,I);LEFT$(SP$,19):NEXT I: RETURN
700 REM
710 COLOUR 1:TX=0:TY=23:GOSUB 900:PRINT SP$
720 TX=0:TY=23:GOSUB 900:RETURN
```

Rutina de evaluación de mano

```
800 REM
810 AC=0:AV=0:TT(PL,1)=0
815 FOR I=1 TO HP(PL)−1
820 IF HD(PL,I,1)=1 THEN AC=AC+1
825 IF HD(PL,I,1)>10 THEN
    TT(PL,1)=TT(PL,1)+10−HD(PL,I,1)
830 TT(PL,1)=TT(PL,1)+HD(PL,I,1)
840 NEXT I
845 IF AC> 0 THEN AV=10
850 TT(PL,2)=TT(PL,1)+AV
852 IF(TT(PL,1)<=21 OR TT(PL,2)<=21) AND HP(PL)>5
    THEN EF=5:RETURN
854 IF HD(PL,1,1)=1 AND HD(PL,2,1)> 10 THEN
    EF=2:RETURN
855 IF HD(PL,2,1)=1 AND HD(PL,1,1)> 10 THEN
    EF=2:RETURN
856 IF TT(PL,1)=21 OR TT(PL,2)=21 THEN EF=3:RETURN
858 IF TT(PL,1)<21 OR TT(PL,2)<21 THEN EF=1:RETURN
860 IF TT(PL,1)>21 AND TT(PL,2)>21 THEN EF=4:RETURN
2300 REM
2305 GOSUB 800
2310 IF TT(PL,1)<>TT(PL,2) OR TT(PL,1)<12 OR
     TT(PL,1)>14 THEN RETURN
2340 GOSUB 700:PRINT"QUEMAR (S/N)";
2345 RESP$=GET$
2347 IF RESP$<>CHR$(13) THEN PRINT RESP$
2350 IF RESP$<>"S" THEN RETURN
2360 HP(1)=1
2370 EP=0:GOSUB 670
2380 FL=0:PL=1:GOSUB 1300:GOSUB 800
2390 FL=0:PL=1:GOSUB 1300:GOSUB 800
2400 GOTO 2300
```

Gama Amstrad CPC

```
 60 fl=0:pl=1:GOSUB 1300:REM repartir naipe a apostador
 70 fl=1:pl=2:GOSUB 1300:REM repartir naipe a la banca
 85 fl=0:pl=1:GOSUB 1300:REM repartir naipe a apostador
 90 fl=0:pl=3:GOSUB 1300:REM repartir naipe a la banca
 95 REM **** turno del apostador ****
100 pl=1
102 GOSUB 2300:REM opcion quemar
```

Dimensionar matrices de manos

```
550 DIM hd(2,5,2),hp(2):REM manos apostador y banca
555 DIM tt(2,2):REM totales marcadores
670 REM **** borrar naipe ****
675 x(pl)=ep:y(pl)=0:tx=ep:ty=0
680 FOR i=1 TO 12:GOSUB 900:PRINT SPACE$(19)
690 ty=ty+1:NEXT i:RETURN
700 REM **** preparar para entrada ****
710 PEN blanco:tx=0:ty=23:GOSUB 900:PRINT SPACE$(39)
720 tx=0:ty=23:GOSUB 900:RETURN
```

Rutina de evaluación de mano

```
800 REM **** evaluar mano ****
810 ac=0:av=0:tt(pl,1)=0
815 FOR i=1 TO hp(pl)−1
820 IF hd(pl,i,1)=1 THEN ac=ac+1
825 IF hd(pl,i,1)> 10 THEN tt(pl,1)=tt(pl,1)+10-hd(pl,i,1)
830 tt(pl,1)=tt(pl,1)+hd(pl,i,1)
840 NEXT i
845 IF ac> 0 THEN av=10
850 tt(pl,2)=tt(pl,1)+av
852 IF (tt(pl,1)<=21 OR tt(pl,2)<=21)AND hp(pl)> 5 THEN
    ef=5:RETURN
854 IF hd(pl,1,1)=1 AND hd(pl,2,1)> 10 THEN ef=2:RETURN
855 IF hd(pl,2,1)=1 AND hd(pl,1,1)>10 THEN ef=2:RETURN
856 IF tt(pl,1)=21 OR tt(pl,2)=21 THEN ef=3:RETURN
858 IF tt(pl,1)< 21 OR tt(pl,2)<21 THEN ef=1:RETURN
860 IF tt(pl,1)> 21 AND tt(pl,2)>21 THEN ef=4:RETURN
2300 REM **** opcion quemar ****
2305 GOSUB 800:REM evaluar
2310 IF tt(pl,1)<>tt(pl,2) OR tt(pl,1)< 12 OR tt(pl,2)> 14
     THEN RETURN
2340 GOSUB 700:PRINT "Quemar (s/n)";
2345 resp$="":WHILE resp$="":resp$=INKEY$:WEND
2347 IF resp$<>CHR$(13) THEN PRINT resp$
2350 IF resp$<> "s" THEN RETURN
2360 hp(pl)=1:REM restablecer puntero mano
2370 ep=0:GOSUB 670:REM borrar naipes
2380 fl=0:pl=1:GOSUB 1300:GOSUB 800:REM repartir naipe a
     apostador
2390 fl=0:pl=1:GOSUB 1300:GOSUB 800:REM repartir naipe a
     apostador
2400 GOTO 2300:REM quemar otra vez?
```

Commodore 64

```
 60 FL=0:PL=1:GOSUB 1300:REM REPARTIR NAIPE A
    APOSTADOR
 70 FL=1:PL=2:GOSUB 1300:REM REPARTIR NAIPE A LA
    BANCA
 85 FL=0:PL=1:GOSUB 1300:REM REPARTIR NAIPE A
    APOSTADOR
 90 FL=0:PL=2:GOSUB 1300:REM REPARTIR NAIPE A LA
    BANCA
 95 REM **** TURNO DEL APOSTADOR ****
100 PL=1
102 GOSUB 2300:REM OPCION QUEMAR
120 GOSUB 2600:REM DOBLAR ETC
550 DIM HD(2,5,2),HP(2):REM MANOS DEL APOSTADOR Y DE
    LA BANCA
555 DIM TT(2,2):REM TOTALES MARCADORES
```

Dimensionar matrices de manos

```
670 REM **** BORRAR VISUALIZACION NAIPE ****
675 X(PL)=EP:Y(PL)=0
680 PRINT CHR$(19);:FOR I=1 TO 12:PRINT
    TAB(EP);LEFT$(SP$,19):NEXT I: RETURN
700 REM **** PREPARAR PARA ENTRADA ****
710 PRINT CHR$(5);:TX=0:TY=23:GOSUB 900:PRINT SP$
720 TX=0:TY=23:GOSUB 900:RETURN
```

Rutina de evaluación de mano

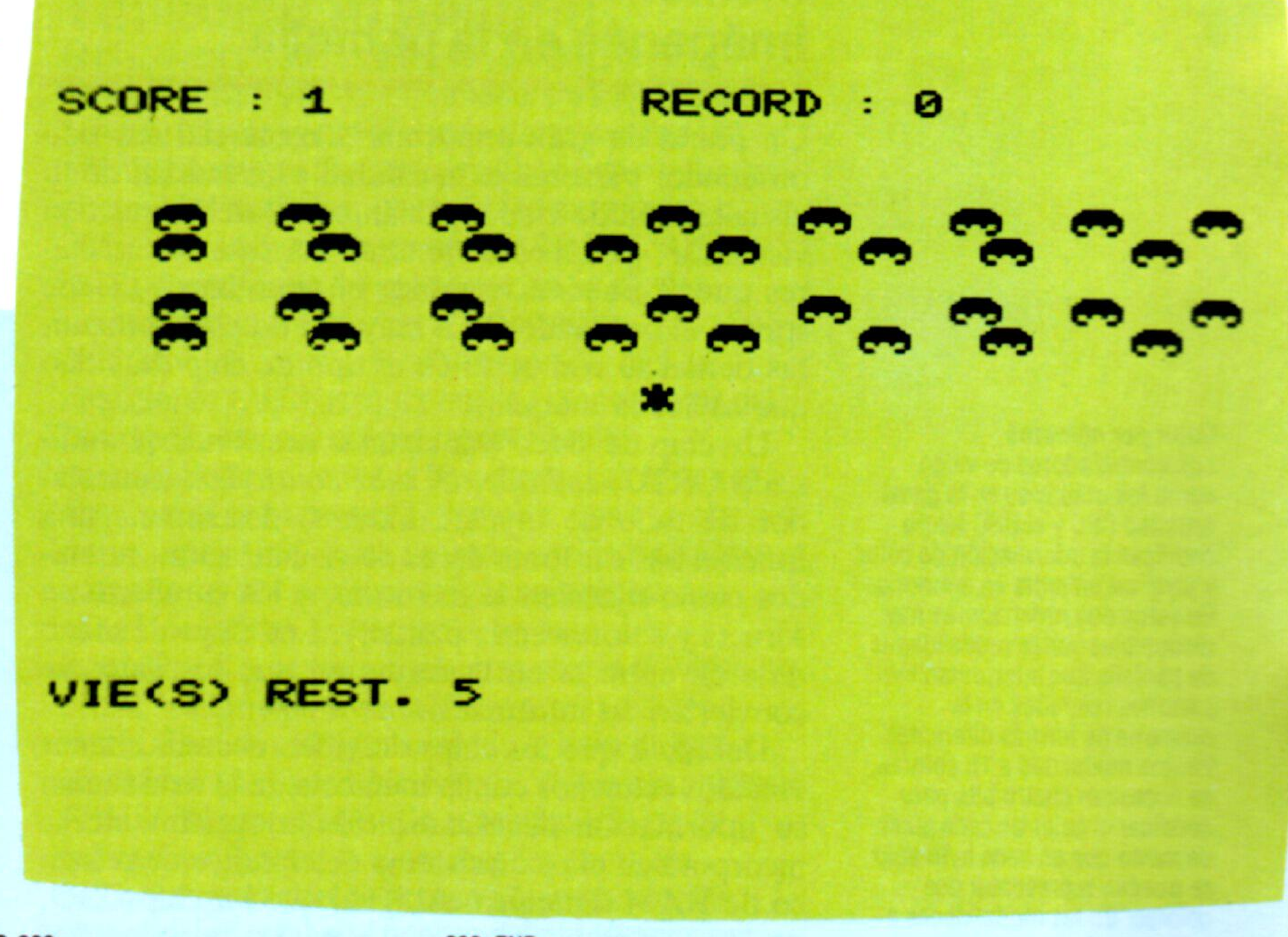
```
800 REM **** EVALUAR MANO ****
810 AC=0:AV=0:TT(PL,1)=0
815 FOR I=1 TO HP(PL)−1
820 IF HD(PL,I,1)=1 THEN AC=AC+1
825 IF HD(PL,I,1)> 10 THEN TT(PL,1)=TT(PL,1)+
    10−HD(PL,I,1)
830 TT(PL,1)=TT(PL,1)+HD(PL,I,1)
840 NEXT I
845 IF AC> 0 THEN AV=10
850 TT(PL,2)=TT(PL,1)+AV
852 IF (TT(PL,1)< =21 OR TT(PL,2)<=21) AND HP(PL)>5
    THEN EF=5:RETURN
854 IF HD(PL,1,1)=1 AND HD(PL,2,1)> 10 THEN
    EF=2:RETURN
855 IF HD(PL,2,1)=1 AND HD(PL,1,1)> 10 THEN
    EF=2:RETURN
856 IF TT(PL,1)=21 OR TT (PL,2)=21 THEN EF=3:RETURN
858 IF TT(PL,1)< 21 OR TT(PL,2)< 21 THEN EF=1:RETURN
860 IF TT(PL,1)> 21 AND TT(PL,2)> 21 THEN EF=4:RETURN
2300 REM **** OPCION QUEMAR ****
2305 GOSUB 800:REM EVALUAR
2310 IF TT(PL,1)<>TT(PL,2) OR TT(PL,1)< 12 OR TT(PL,1)>
     14 THEN RETURN
2340 GOSUB 700:PRINT"QUEMAR (S/N)";
2345 GET RESP$:IF RESP$="" THEN 2345
2347 IF RESP$<> CHR$(13) THEN PRINT RESP$
2350 IF RESP$<> "S" THEN RETURN
2360 HP(1)=1:REM RESTABLECER PUNTERO MANO
2370 EP=0:GOSUB 670:REM BORRAR NAIPES
2380 FL=0:PL=1:GOSUB 1300:GOSUB 800:REM REPARTIR
     NAIPE A APOSTADOR
2390 FL=0:PL=1:GOSUB 1300:GOSUB 800:REM REPARTIR
     NAIPE A APOSTADOR
2400 GOTO 2300:REM QUEMAR OTRA VEZ?
```



Cangrejos

Las leyes de la selección natural y de la supervivencia de los más aptos enmendadas por la informática. Esta versión de este original juego ha sido escrita para los ordenadores MSX

Usted debe ayudar a una pobre tortuga a volver al mar, evitando a los voraces cangrejos que deambulan por la playa. Cada tortuga que alcance su propósito le proporciona 1 punto. Dispone de cinco itinerarios para intentar marcar su puntuación máxima. Emplee las teclas de control del cursor para avanzar y para retroceder.




```
10 REM *************
20 REM * CANGREJOS *
30 REM *************
40 SCREEN 0,0
50 DEFINT A-Z
60 KEY OFF
70 WIDTH 39
80 GOSUB 990
90 GOSUB 1130
100 GOSUB 840
110 LOCATE 0,20,0
120 PRINT "VIDAS RESTANTES.";NP;
130 A$=RIGHT$(A$,1)+LEFT$ (A$,38)
140 B$=RIGHT$(B$,38)+LEFT$(B$,1)
150 LOCATE 0,X1,0
160 PRINT A$;
170 LOCATE 0,X2,0
180 PRINT B$;
190 LOCATE 0,X3,0
200 PRINT A$;
210 LOCATE 0,X4,0
220 PRINT B$;
230 D$=INKEY$
240 PY=PY+(STICK(0)=1)-(STICK(0)=5)
250 IF PY>10 THEN PY=10
260 IF PY=2 THEN 370
270 C=VPEEK(PX+PY*40+1)
280 IF C<>32 AND C<>128 THEN 550
290 LOCATE PX,YP,0
300 PRINT N$;
310 LOCATE PX,PY,0
320 PRINT P$;
330 YP=PY
340 T=T+1
350 IF T>500 THEN 660
360 GOTO 110
370 LOCATE PX,YP,0
380 PRINT N$;
390 LOCATE PX,PY,0
400 PRINT P$;
410 BEEP
420 FOR I=1 TO 200
430 NEXT I
440 LOCATE PX,PY,0
450 PRINT N$;
460 PY=10
470 YP=PY
480 S=S+1
490 LOCATE 0,0,0
500 PRINT "PUNTUACION:";S:
510 LOCATE 19,0
520 PRINT "RECORD :";R·
530 GOSUB 1040
540 GOTO 110
550 NP=NP-1
560 LOCATE PX,YP,0
570 PRINT N$;
580 LOCATE PX,PY,0
590 PRINT CHR$(128);
600 GOSUB 1090
610 IF NP=0 THEN 660
620 PY=10
630 YP=PY
640 GOSUB 1040
650 GOTO 110
660 CLS
670 IF S>R THEN R=S
680 IF T<500 THEN 710
690 LOCATE 10,8,0
700 PRINT "** TIEMPO TRANSCURRIDO**";
710 LOCATE 10,12,0
720 PRINT "PUNTUACION:";S;
730 LOCATE 10,16,0
740 PRINT "RECORD :";R;
750 LOCATE 10,20
760 PRINT "OTRA ?";
770 IF INKEY$<>"" THEN 770
780 D$=INKEY$
790 IF D$="" THEN 780
800 IF D$<>"N" AND D$<>"n" THEN 100
810 CLS
820 LOCATE Q,0,1
830 END
840 CLS
850 COLOR 1,11
860 P$=CHR$(128)
870 N$=CHR$(32)
880 S=0
890 NP=5
900 PX=19
910 PY=10
920 YP=PY
930 X1=4
940 X2=5
950 X3=7
960 X4=8
970 T=0
980 RETURN
990 FOR I=1 TO 39
1000 READ A
1010 A$=A$+CHR$(A)
1020 NEXT I
1030 B$=A$
1040 X=RND(1)*35+2
1050 A$=RIGHT$(A$,X)+LEFT$(A$,39-X)
1060 RETURN
1070 DATA 32, 129, 130, 32, 32, 129, 130, 32, 32,
     32, 129, 130, 32, 32, 32, 32, 32, 129, 130, 32
1080 DATA 32, 32, 129, 130, 32, 32, 32, 129, 130,
     32, 32, 129, 130, 32, 32, 129, 130, 32, 32
1090 PLAY "T10003G1604C203G804C16E2T150C
     8E16G8F+16f8D+16E8C1603A8G1604C4C32E16
1100 FOR I=1 TO 4000
1110 NEXT I
1120 RETURN
1130 FOR I=0 TO 23
1140 READ A
1150 VPOKE 3072+I,A*4
1160 NEXT I
1170 RETURN
1180 DATA 12,45,63,30,30,63,45,0
1190 DATA 7,15,31,31,18,16,12,0
1200 DATA 56,60,62,62,18,2,12,0
```

© SYBEX

El chip de video

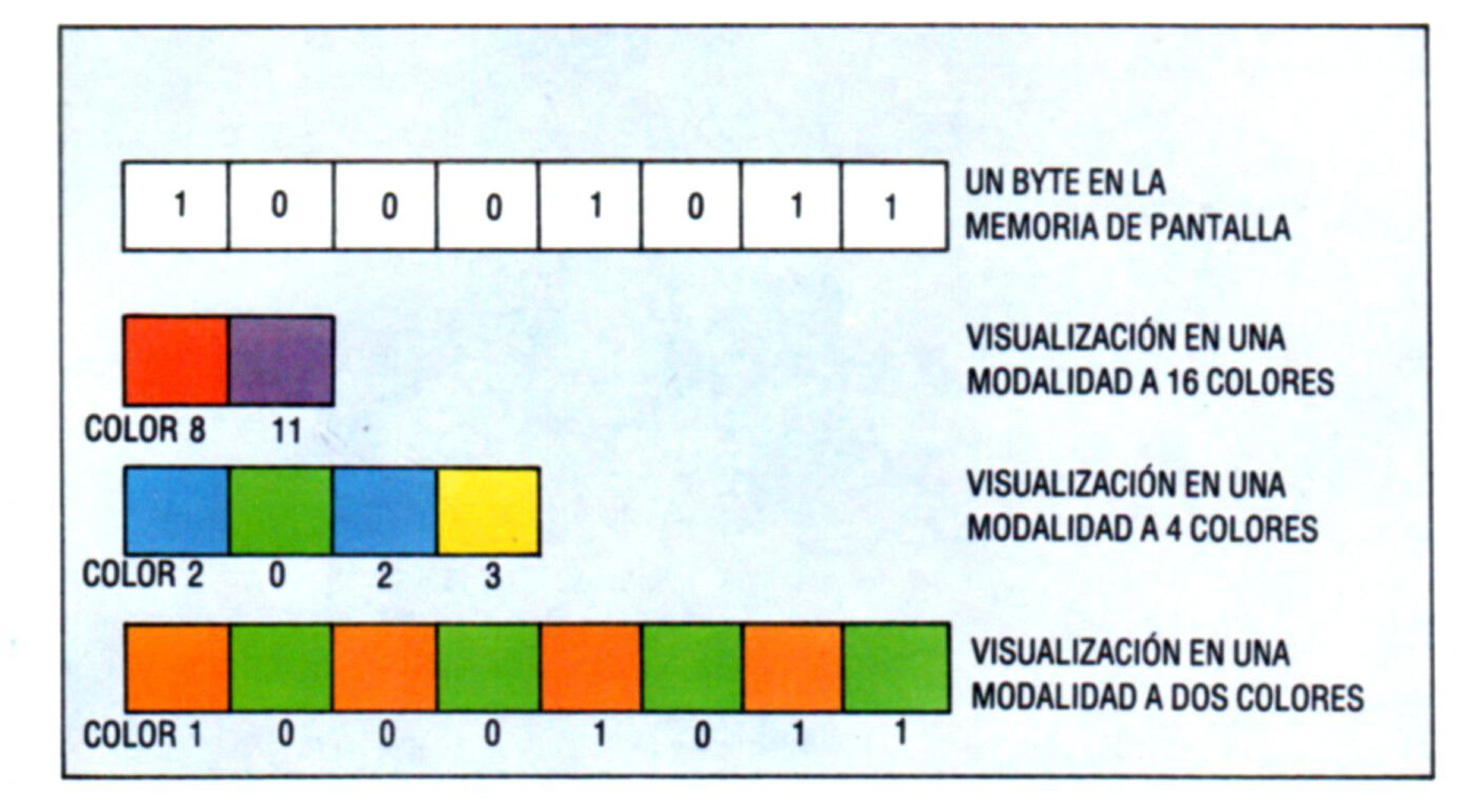
Veamos ahora de qué manera los patrones binarios de la memoria se convierten en imágenes en la pantalla

Un punto de gran importancia comercial en todo ordenador personal es la calidad y diversidad de su visualización de video. ¿Cuántos colores se pueden visualizar? ¿Cuál es la cantidad máxima de caracteres que se pueden visualizar en una línea? ¿Tiene sprites el ordenador? La mayoría de estas preguntas tienen su respuesta en el tipo de chip de video que utilice la máquina.

Un chip de video (en algunas ocasiones denominado CRTC: *cathode-ray tube controller*: controlador de tubo de rayos catódicos) desempeña una función básica: toma datos de visualización, retenidos como bytes en la memoria, y los convierte en formas y colores en la pantalla. Los chips de video difieren entre sí en la forma en que los datos se convierten en información de imagen.

Debido a que los chips de video necesitan tener vínculos estrechos con la memoria de la cual toman su información de visualización, normalmente se incorpora en ellos un sistema de circuitos de refresco de RAM dinámica. Al igual que los chips PIO, de los que hablamos en el capítulo anterior, los chips de video poseen registros internos programables que controlan sus funciones. La programación directa del chip la suele realizar el sistema operativo del ordenador, pero se pueden conseguir efectos de video especiales si el mismo usuario programa el chip de video.

Color por números
Los controladores de video como los utilizados en la gama Amstrad CPC y el BBC Micro codifican la información de color y pixel *on/off* junta en memoria. En estos dos ordenadores hay disponibles varias modalidades de pantalla que interpretan los patrones retenidos en la memoria de formas diferentes. En una modalidad a 16 colores, se necesitan cuatro bits para codificar el color de cada pixel; de modo que en cada byte sólo se pueden representar dos colores. En las modalidades a cuatro y dos colores, sólo se necesitan dos y un bit, respectivamente, para el código de color de un pixel. En estas modalidades un byte puede representar cuatro u ocho pixels



El primer tipo de chip de video que analizaremos retiene todos sus datos de visualización y de color juntos en la memoria, haciendo que la tarea de conversión a una imagen resulte mucho más sencilla. Éste es el tipo de visualización que utilizan el BBC Micro, el Acorn Electron y la gama Amstrad. Tales chips normalmente soportan una cantidad de modalidades de visualización diferentes en las que varían el tamaño de los caracteres y el número de colores disponibles. El diagrama ilustra cómo se interpreta un único byte de la memoria en distintas modalidades de pantalla. Si bien este tipo de disposición simplifica el trabajo del chip de video, y es fácil mezclar texto y gráficos en alta resolución, la desventaja es la gran cantidad de memoria requerida para retener una pantalla de información. Una visualización de 160 por 256 pixels con 16 colores, por ejemplo, requiere 20 Kbytes de memoria.

Cuando el costo de la memoria era elevado, tales métodos de visualización no eran factibles y se empleaba un método de visualización alternativo, que aún está presente en el Commodore 64. En vez de retener cada pixel de información de visualización y color de forma explícita en la memoria, este segundo tipo de chip de video utiliza un sistema simplificado para visualizar gráficos de caracteres solamente. En tal visualización, cada posición de carácter corresponde a un byte de la memoria. Cada byte retiene tan sólo un código de carácter que el chip de video utiliza para acceder a una tabla de ROM de las verdaderas definiciones de caracteres de ocho por ocho pixels. De este modo, una visualización de caracteres de 40 por 25 requiere menos de un Kbyte.

Las implementaciones en color de este tipo de sistema emplean otra zona de 1 000 bytes para retener los datos de color; cada byte retiene un código de color para la posición del carácter correspondiente.

El sistema de visualización de video empleado por el BBC Micro (y otros) aún ha de utilizar definiciones de caracteres de ROM, pero el sistema operativo accede a las mismas cuando ha de escribir un carácter en la memoria de pantalla. El proceso de buscar la definición y realizar las manipulaciones necesarias para codificar la información integral de color se lleva a cabo en la etapa de escribir en la memoria de pantalla, y no en la etapa de conversión de memoria a imagen.

Las visualizaciones de video basadas en este segundo método normalmente poseen una segunda modalidad en alta resolución en la cual los bits retenidos en la memoria guardan una correspondencia directa con los pixels de la visualización final. Este tipo de visualización de gráficos se conoce como *bit mapping* (mapa de bits). La información de color no se codifica en los bits de la visualización (como en el primer sistema descrito), sino que se retiene del mismo modo que para la visualización de caracteres solamente. Es decir, una zona de RAM de 1 000 bytes en la que cada byte retiene información de color para una zona de la pantalla de ocho por ocho pixels.

En el Commodore 64, esto permite gráficos en alta resolución, pero no se puede mezclar texto directamente a menos que el usuario escriba sus propias rutinas para escribir en el mapa de bits las verdaderas definiciones de caracteres. Resulta interesante que sea éste el enfoque adoptado para el Spectrum, que almacena su pantalla como un simple mapa de bits y utiliza una RAM de color separada para controlar cada sección de ocho por ocho pixels de la pantalla. En el Spectrum se pueden mezclar caracteres y gráficos en alta resolución por-



que el sistema operativo escribe las verdaderas definiciones de caracteres en el mapa de bits.

El chip de video controla la visualización de gráficos sprite (en caso de que exista dicha facilidad). Por lo general, los sprites se definen mediante imágenes de bits en RAM, y sus posiciones en la pantalla, así como otros atributos, tales como color y tamaño, los controlan registros internos del chip. Dado que se utiliza un único chip para manipular gráficos normales y sprites, los chips de video como el VIC del Commodore 64 se pueden programar para generar interrupciones cuando los sprites chocan con otros sprites o con datos del fondo.

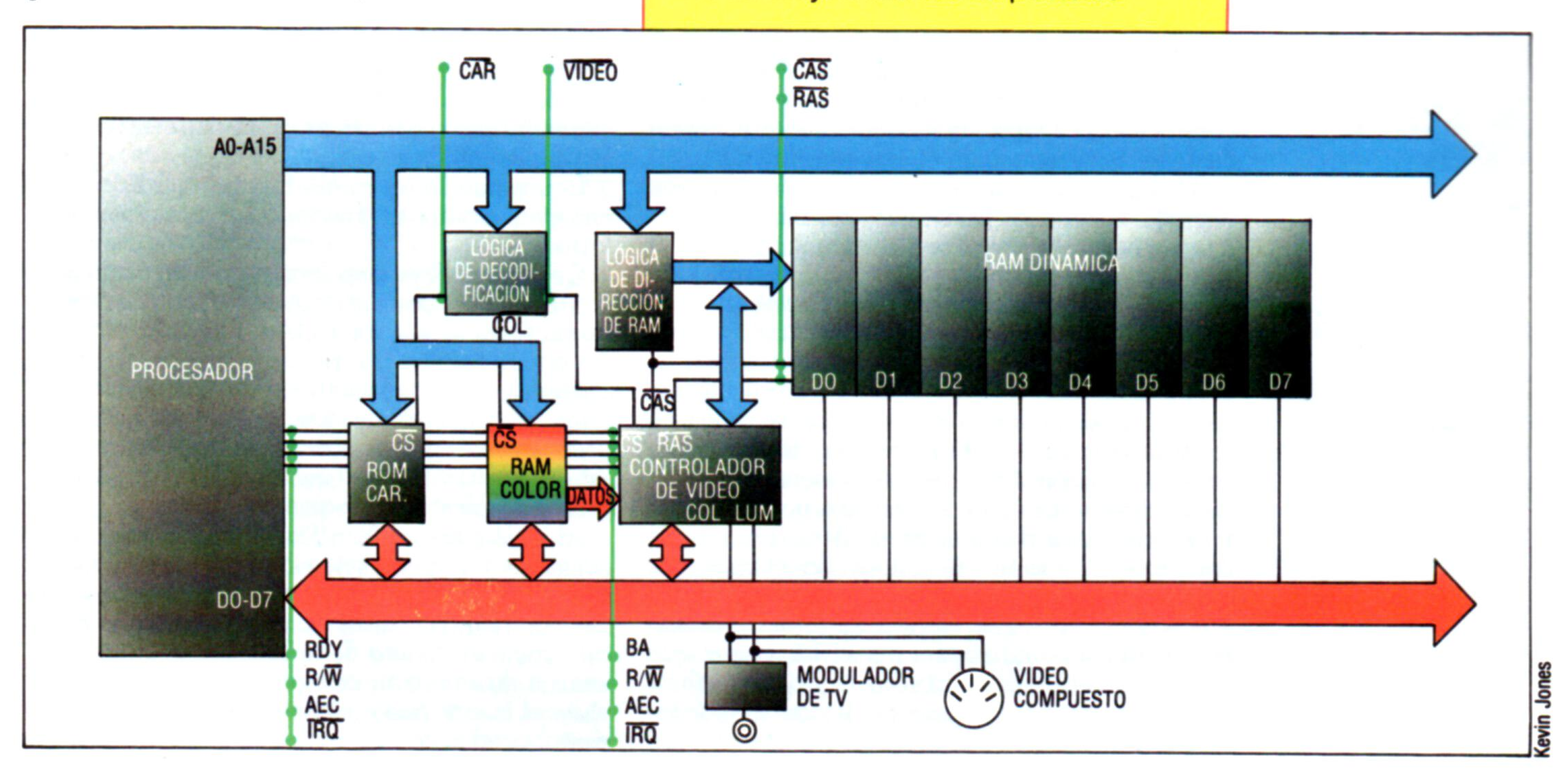
Los dos tipos de chip de video que estamos analizando aquí soportan el desplazamiento fuera de la pantalla de formas diferentes. El segundo tipo realiza el desplazamiento a través de software. Puesto que el número de bytes que componen la visualización en pantalla es reducido, el traslado de los datos de pantalla a posiciones nuevas se puede efectuar con rapidez en código máquina. El alto número de bytes necesarios para retener la pantalla en el primer método lleva a que el desplazamiento por software sea demasiado lento. Por consiguiente, los chips de video basados en este sistema incorporan un registro de "comienzo de pantalla", que retiene la dirección del principio de la memoria de pantalla. Por lo tanto, el desplazamiento se puede conseguir con sólo cambiar la dirección de este registro, y con una manipulación cuidadosa se puede utilizar este método para desplazar la pantalla hacia arriba, hacia abajo, a izquierda o a derecha.

El tipo de chip de video empleado en el BBC Micro tiene mucho menos trabajo que hacer que, por poner un ejemplo, el chip VIC del Commodore 64. En consecuencia, puede realizar sus accesos a la memoria durante la fase del ciclo de reloj, cuando el procesador propiamente dicho no accede a la memoria. Este chip de video, por lo tanto, no reduce la velocidad del procesador. Además de utilizar el chip de video, el BBC incorpora una ULA diseñada a medida que proporciona la temporización para la totalidad del sistema, calcula las relaciones entre colores lógicos y físicos y proporciona salida de video RGB.

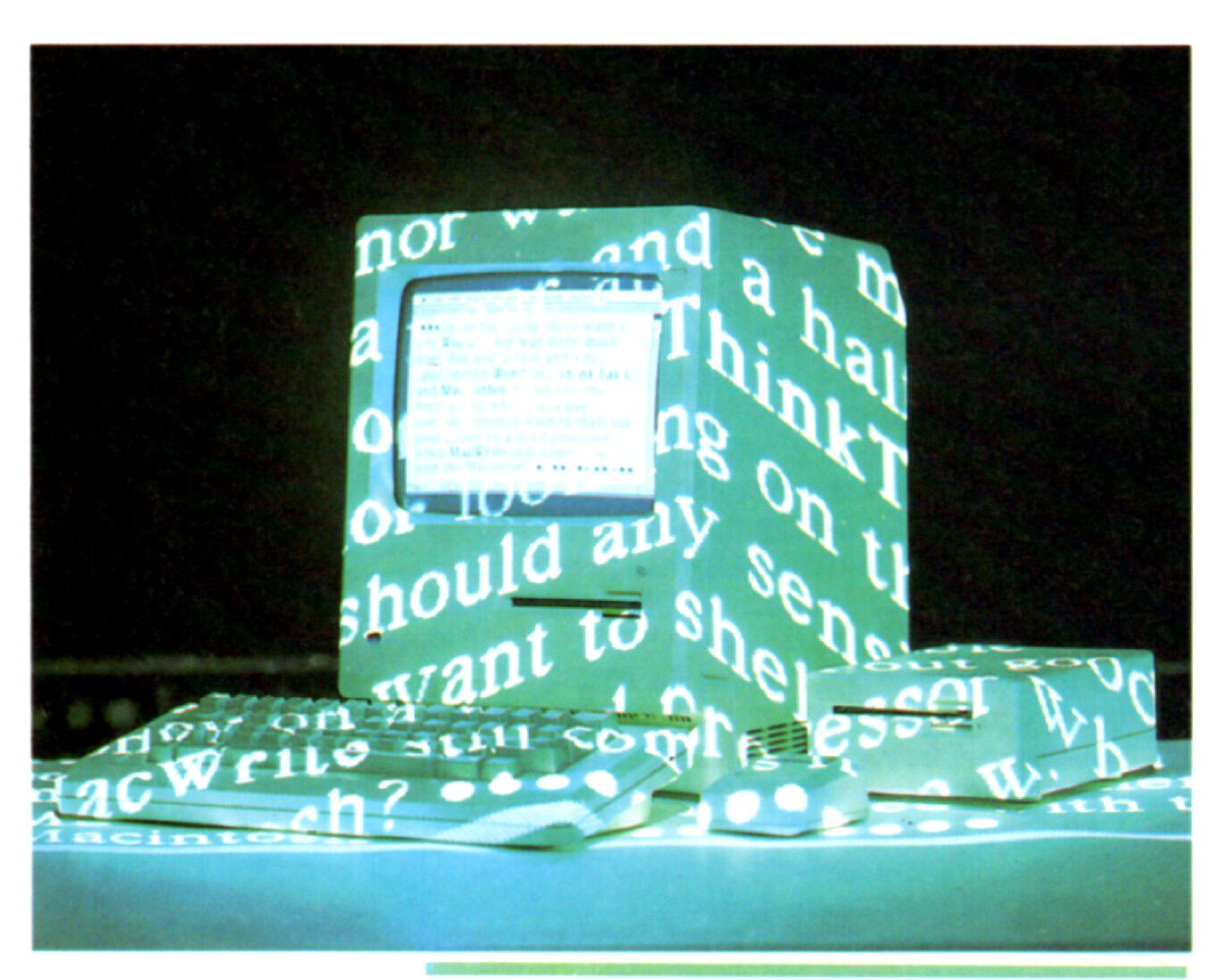
El chip VIC necesita efectuar varios accesos a memoria. Algunos, como el refresco de la RAM y las búsquedas de datos de caracteres (en el BBC Micro), son invisibles para el procesador, porque se pueden realizar en la fase alternada del reloj del sistema. Algunas funciones necesitan datos a una velocidad mayor que ésta y las operaciones del procesador han de ser suspendidas temporalmente mientras el chip de video realiza sus accesos a la memoria.

Tomando la imagen

El diagrama muestra cómo se conecta con el sistema el chip VIC del Commodore 64. El VIC se conecta directamente al bus de datos y utiliza las mismas líneas de dirección multiplexadas utilizadas por la RAM dinámica, que también se refresca utilizando las líneas de control CAS y RAS. Algunas de las operaciones del chip de video se pueden llevar a cabo sin perturbar al procesador. Asimismo, el control de habilitación del bus de direcciones (AEC) asegura que los manejadores del bus de direcciones del procesador estén desconectados durante la fase de reloj en la que el chip realiza sus accesos a memoria. Los accesos a las zonas de 1000 bytes de visualización de RAM y datos de sprites se debe realizar más rápido que cada fase alterna, y, por tanto, es necesario inhabilitar el procesador mientras el chip de video "roba" ciclos de memoria para acceder a estas zonas. Esto se consigue uniendo la línea de bus disponible (*bus available*: BA) del VIC con la patilla READY del procesador. Se conceden tres ciclos más de reloj para que el procesador complete cualquier operación de acceso a la memoria. Sin embargo, en la fase de acceso del procesador del cuarto ciclo de reloj, AEC permanecerá *low* para permitirle al VIC el acceso a la memoria durante esta fase. El mantenimiento de la visualización de video disminuye la velocidad del procesador



Programa atípico



Sobre él ya se ha escrito todo
Apple se ha enfrentado con una enorme reticencia por parte de los destinatarios a la hora de querer imponer el Macintosh como máquina de gestión seria. Parte del problema ha residido, sin duda, en la ausencia de un exhaustivo paquete para tratamiento de textos, que es un prerrequisito para las aplicaciones de gestión. No obstante, han aparecido varios programas para tratamiento de textos, si bien sólo son un puñado en comparación con los centenares que existen a la venta para el IBM PC

Profusión de iconos
Al igual que todos los paquetes escritos para el Macintosh, el *MacWrite* hace un uso intensivo de los iconos para ejecutar instrucciones posicionadas a lo largo del perímetro de la pantalla. Muchos de éstos (como la barra de desplazamiento) se utilizan comúnmente en numerosos programas para el Macintosh, mientras que otros son exclusivos del *MacWrite*

Conozcamos el "MacWrite", el paquete de tratamiento de textos producido por Apple para el Macintosh

Contrastando con los paquetes que hemos visto en anteriores capítulos, que se basan en menús y caracteres de control, *MacWrite*, el programa de tratamiento de textos empaquetado con el Macintosh, se basa en gran medida en los iconos y las ventanas controladas por ratón, que ahora ya se esperan en cualquier programa para el Macintosh.

Al cargar *MacWrite*, al usuario se le presenta una pantalla de apertura, compuesta por una gran ventana en la cual se visualizará el documento. Arriba hay una serie de iconos y una regla, y arriba de todo hay una barra de menú compuesta por seis títulos. Debajo de ésta se halla la barra de títulos, donde aparece el nombre del documento actual.

A diferencia de la mayoría de los otros paquetes, en los que el cursor se desplaza con la ayuda de teclas de control, *MacWrite* lo manipula con el controlador de ratón. En otros procesadores de textos, el cursor se utiliza para señalar una posición en el texto que indica dónde se ha de llevar a cabo una determinada opción. Se puede posicionar, por ejemplo, para insertar texto o crear un bloque. En *MacWrite* el cursor se emplea de forma idéntica, pero también se utiliza para mover los iconos, que alteran los controles de tabulación y de márgenes, así como para bajar y seleccionar las entradas de los menús.

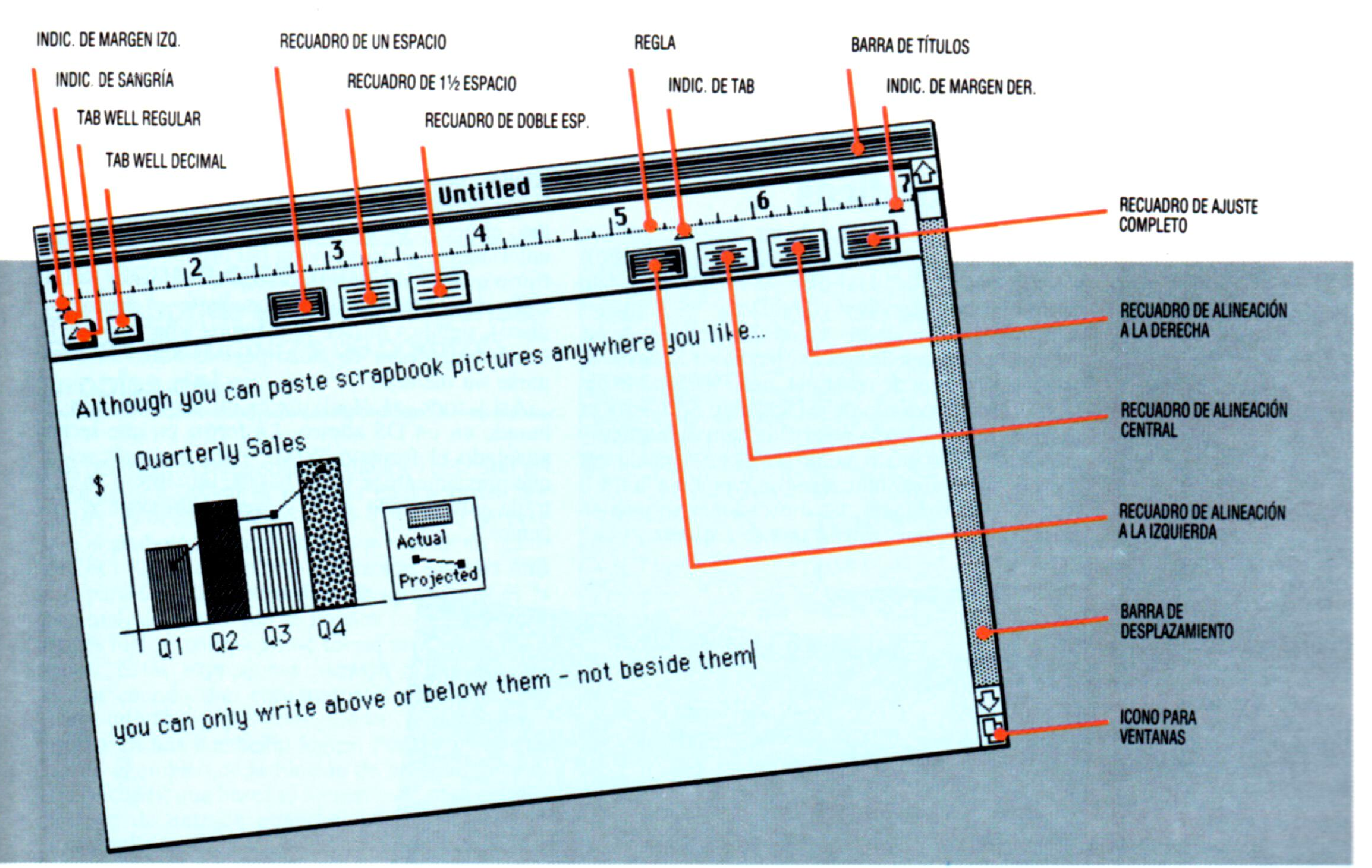
Al bajar un menú se visualizan varias opciones diferentes, pero no todas ellas son utilizables de inmediato. Las que sí están disponibles se indican en letras negras, mientras que las otras están sombreadas en gris. Con frecuencia el menú consiste en un cierto número de opciones preestablecidas, habiendo una marca junto a la seleccionada actualmente. La selección de otra opción preestablecida trasladará la marca hasta esa selección, indicando que ésa es la opción actual por defecto.

El menú File (de archivos) contiene las instrucciones DOS que permiten abrir y cerrar archivos, además de guardarlos bajo el nombre de archivo actual o uno distinto. También se incluye en este menú la instrucción Print (imprimir).

El segundo de los menús listados es Edit (editar), que proporciona instrucciones para manipular el texto.

Quizá para el principiante la más importante de éstas sea Undo (deshacer), que borrará todas las acciones llevadas a cabo desde que se pulsó el botón del ratón. Tal vez esto parezca drástico, pero muchos errores se producirán a resultas de selecciones erróneas con el ratón. De modo que, utilizando Undo, se perderá muy poco del trabajo valioso. En este menú también se incluyen las instrucciones para movimiento de bloques.

Al igual que con *WordStar*, los bloques se crean delimitando el texto seleccionado. Tras colocar el cursor ya sea al comienzo o al final del texto y pulsando el ratón, el cursor cambiará a un icono conocido como el "punto de inserción", que permitirá insertar texto a partir de esa posición. Si usted desplaza el cursor hacia arriba o hacia abajo de este punto, el texto correspondiente aparecerá en video



invertido, lo que en realidad marca el bloque. Una segunda pulsación del ratón establecerá entonces el otro extremo del bloque.

Una vez marcado el bloque, se pueden realizar con él varias operaciones diferentes. Retornando al menú Edit, por ejemplo, se puede seleccionar la opción Cut (cortar), que coloca al bloque en un "tablero de chinchetas". Éste es un buffer que almacena el bloque antes de efectuar operaciones en él. Para contemplar el contenido de este buffer, se selecciona la opción Show Clipboard (mostrar "tablero de chinchetas") del menú Edit. Estableciendo otro punto de inserción y retornando al menú Edit, usted puede seleccionar las opciones Copy (copiar) o Paste (pegar), que le permiten ya sea copiar o transferir bloques en esa posición. Observe que todos los bloques implicados en estas operaciones se reformatearán automáticamente.

El menú Search (buscar) contiene la instrucciones Find/Replace (hallar/reemplazar) y es idéntico al implementado en la mayoría de los paquetes de tratamiento de textos.

Muchos procesadores de textos permiten utilizar diferentes tipos de letra en un documento dado, pero la mayoría de ellos se limitan a los que utilizan para dar énfasis, tales como cursiva y negrita. Además, dado que la mayoría de los paquetes se limitan a utilizar las matrices de caracteres que ya están retenidas en la ROM del sistema operativo del ordenador, por lo general es imposible ver exactamente cómo aparecerá el documento hasta que se lo ha impreso.

MacWrite, sin embargo, contiene muchos tipos de letra diferentes, añadiéndosele más a medida que aparece cada nueva versión de software. Asimismo, debido a que la pantalla de textos de Macintosh es por mapa de bits en lugar de tener la disposición usual de celdas de caracteres, estos tipos de letra se pueden visualizar en la pantalla. No obstante, a pesar de esta facilidad, el sistema no es perfecto. Muchos de los tipos de letra, como el London (de estilo gótico), son casi ilegibles.

Esta facilidad de mapa de bits se utiliza con resultados más satisfactorios en el menú Style (estilo), una serie de opciones que proporcionan las formas de énfasis y efecto más usuales, tales como letras subrayadas y en negrita. Éstas se pueden imprimir en pantalla para poder apreciar su efecto antes de enviar el documento a la impresora.

La mayor parte de las funciones de trazado de página del *MacWrite* están retenidas en el menú Format (formatear), si bien los verdaderos formatos de tamaño de página están retenidos en el menú File. Se pueden establecer cabeceras y pies de página (títulos estándares que aparecerán arriba y abajo de cada página impresa), así como el número de página, hora y fecha. Los tres últimos se visualizan como iconos y se los puede "arrancar" de su propia barra de menú y colocar en cualquier sitio en la zona de cabecera o pie.

Debajo de la regla, en la parte superior de la pantalla, hay una serie de pequeños iconos de flecha que se pueden arrastrar a lo largo para establecer los márgenes, sangría de párrafos y ajustes Tab. El documento se restablecerá automáticamente cuando se modifique algún parámetro. Dos recuadros situados debajo y a la derecha de la regla son los Tab wells, a partir de los cuales usted puede obtener otros iconos y ponerlos en la regla. A la derecha de éstos hay varios iconos de página, cada uno

MACWRITE

Desplazamiento de palabras



El desplazamiento de palabras y la alineación automática están totalmente implementados.

Movimiento de bloques



En un documento, el texto y los gráficos se pueden desplazar y posicionar en cualquier punto.

Ayuda en pantalla

No tiene.

Pantalla de 80 columnas



Sólo posee pant. de 60 columnas en mod. por defecto; el texto con caracteres de matriz de 9 puntos (el tamaño de letra más pequeño) permite 80.

Contador de palabras

En el *MacWrite* no se proporciona ninguna facilidad para contar las palabras.

Buscar/reemplazar



Permite buscar y alterar hasta 44 caracteres en la ventana actual.

WYSIWYG



Mostrar varios tipos de letra y diagramas en pantalla es una de sus características fundamentales.

Facil. para correspondencia

No tiene.

Verificador de ortografía

No tiene, aunque hay a la venta utilidades producidas por fabricantes independientes.

Tipos de letra disponibles



Implementa una amplia variedad de tipos de letra, si bien el número de éstos varía de una versión a otra.

Unión de archivos



A pesar de que algunas facilidades (como Cortar y Pegar) se pueden transportar entre documentos, no existe una facilidad de unión de archivos como tal.

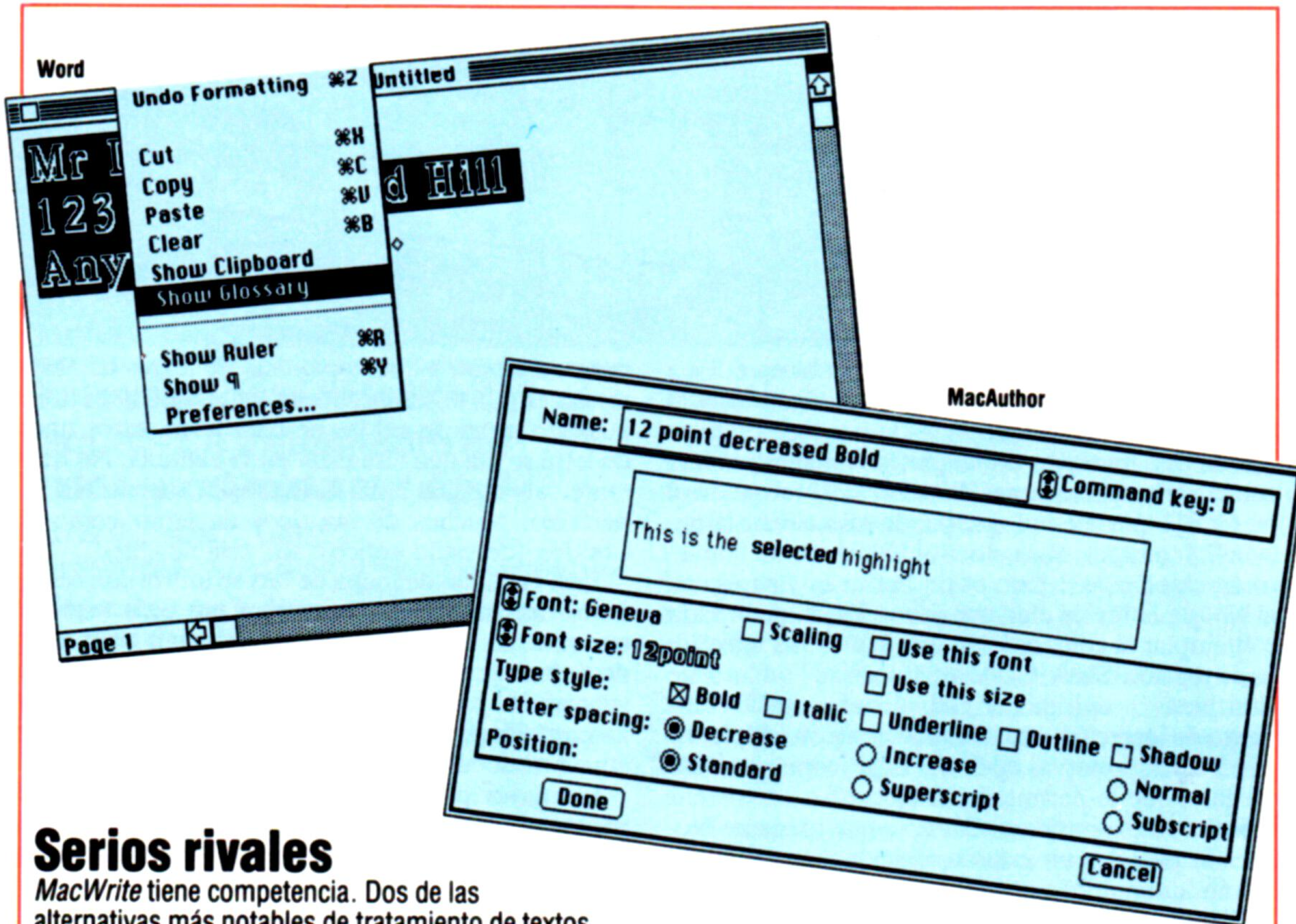
de los cuales posee líneas que indican sus funciones. Los primeros tres corresponden a separación de una línea, de una línea y media y de dos líneas, mientras que los cuatro restantes al otro extremo de la barra indican el ajuste de línea.

Texto y gráficos

El Macintosh permite combinar textos y gráficos en la página y visualizarlos en la pantalla. *MacWrite* soporta esta facilidad, si bien las imágenes se han de dibujar con *MacPaint* y *MacDraw*, los paquetes artísticos empaquetados en el Macintosh. A las ilustraciones se puede acceder mediante la facilidad Scrapbook (álbum de recortes), que forma parte del escritorio incorporado del Macintosh. Scrapbook es un área de la memoria reservada para almacenar y transferir tanto texto como gráficos a cualquier lugar de un documento. Además, debido a la flexibilidad de la máquina, las ilustraciones se pueden ajustar a cualquier tamaño que se requiera.

Si bien el *MacWrite* es un paquete sumamente sencillo de aprender y utilizar, así y todo posee sus inconvenientes. En primer lugar, el programa deja poco espacio de memoria disponible para texto en el Macintosh estándar. Esto podría resultar difícil de creer considerando que la máquina tiene 128 Kbytes, pero el sistema operativo y la pantalla por mapa de bits ocupan muchísimo espacio. Asimismo, el hecho de que el texto del *MacWrite* resida enteramente en la memoria del ordenador (al contrario que en algunos procesadores de textos avanzados, que guardan continuamente el texto en disco), significa que en la máquina sólo se pueden escribir alrededor de cinco páginas antes de quedarse sin memoria.

Así y todo, el *MacWrite* es un paquete inusual basado en un OS atípico. La forma en que se ha adaptado el formato WIMP para una aplicación que previamente se había basado por entero en entrada por teclado es a la vez interesante e ingeniosa.



Serios rivales

MacWrite tiene competencia. Dos de las alternativas más notables de tratamiento de textos para el Macintosh son *Word*, de Microsoft, y *MacAuthor*, de Icon Technology. Ambos pretenden solucionar el problema central del *MacWrite*: su incapacidad para tratar más de unas pocas páginas de texto. A primera vista, ambos paquetes parecen similares al *MacWrite*, con instrucciones activadas por iconos y una pantalla rodeada por las barras de desplazamiento y títulos. No obstante, contienen facilidades adicionales que hacen del Macintosh una proposición mucho más seria para intentar el tratamiento de textos profesional.

Word contiene varias facilidades destinadas a permitir mayor flexibilidad y potencia, permitiendo manipular hasta cuatro documentos en pantalla al mismo tiempo. También ofrece un tipo mejorado de Glosario, en lugar de depender de la facilidad Scrapbook, que tiene sus limitaciones.

Mientras que *Word* está pensado como sistema de tratamiento de textos de gestión para el Macintosh, *MacAuthor*, como su nombre sugiere, está dirigido a los escritores. Se ha desarrollado para permitir la mayor flexibilidad posible en el uso de caracteres. El usuario puede crear caracteres nuevos, por ejemplo, combinando los ya existentes en el juego de caracteres y alterando el "estilo" del texto a su voluntad. Debido a esta mayor flexibilidad, familiarizarse con *MacAuthor* es un poco más difícil que hacerlo con *Word* o *MacWrite*. A pesar de esto, a muchos escritores este esfuerzo adicional les resultará rentable



Estudio estructural

Llegados a este punto, nos corresponde examinar algunas de las estructuras de control propias del c

Los operadores relacionales ordinarios del c son los mismos que los de la mayoría de los otros lenguajes, es decir, <, <= –>, >= –, pero observe que el símbolo de igualdad es == y el de no igualdad es !=. Los conectores lógicos son && para AND e ll para OR. La precedencia de operadores es la habitual, de modo que se pueden construir expresiones lógicas complejas de forma muy parecida al BASIC. Estas expresiones asumen un significado mayor cuando uno recuerda que una asignación posee un valor y, por consiguiente, se puede comprobar en una condición lógica. Por ejemplo, mediante el empleo de la función de biblioteca estándar getchar(), que busca el siguiente carácter del dispositivo de entrada estándar, podemos construir una condición:

```
contador_car(long_línea_máx – 1&&
    (c=getchar())!=/n'&&c!=EOF
```

Esta condición comprobará primero una cantidad máxima de caracteres y luego, de no haberse alcanzado esta cantidad, buscará el siguiente carácter y comprobará si se trata de un final de línea o un final de archivo, todo ello en una expresión lógica.

Otra característica es que las expresiones lógicas también poseen valores; falso es cero y verdadero es uno (de hecho, todo valor que no sea cero se tomará como verdadero). De modo que, si es necesario, se pueden utilizar expresiones lógicas en los cálculos. Por el contrario, los valores numéricos simples se pueden probar directamente sin utilizar ningún operador. El operador de negación unario ! se puede colocar delante de cualquier valor entero y lo cambiará por cero si no es cero, o por uno si es cero. De modo, entonces, que la comprobación:

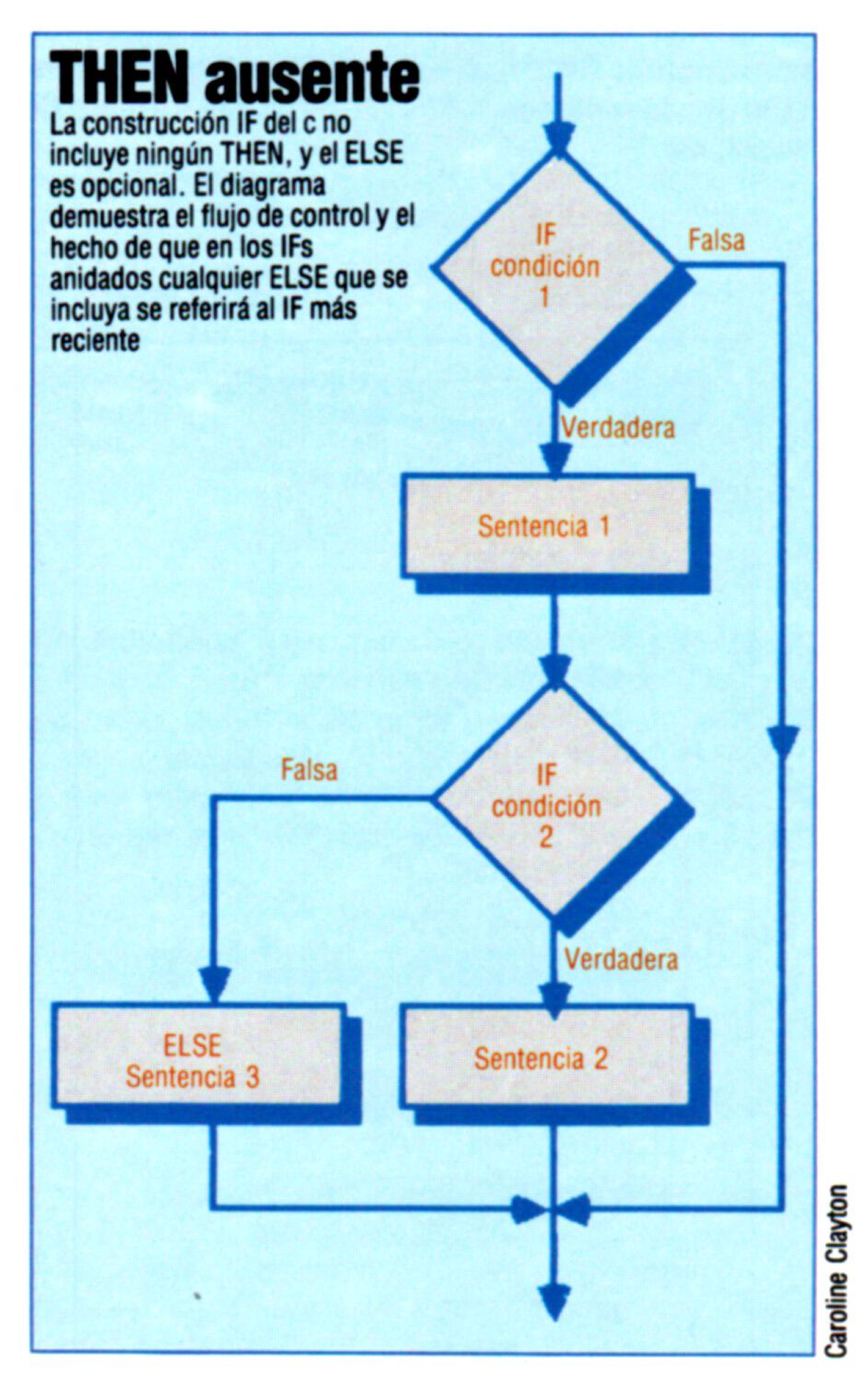
```
if(!valor_ent)
```

equivale a:

```
if (valor_ent==0)
```

Además del conjunto normal de operadores lógicos, el c posee cierto número de operadores lógicos aplicables a nivel de bit que se pueden utilizar sólo en tipos integer o char cualesquiera para proporcionar la manipulación de bits que ofrecen la mayoría de los ensambladores. Éstos son:

&	AND
l	OR inclusivo
∧	OR exclusivo
<<n	donde n es un entero, desplaza n bits hacia la izquierda
>>n	donde n es un entero, desplaza n bits hacia la derecha
~	complemento a uno



THEN ausente
La construcción IF del c no incluye ningún THEN, y el ELSE es opcional. El diagrama demuestra el flujo de control y el hecho de que en los IFs anidados cualquier ELSE que se incluya se referirá al IF más reciente

Caroline Clayton

A modo de ejemplo, a menudo se emplea el operador & para enmascarar ciertos bits de modo que se puedan comprobar uno o varios en una palabra. Para comprobar si el bit 3 está establecido en uno en un único byte, c, podemos usar:

```
if(c & 0x08)
```

Observe la notación 0x, que precede a una constante hexa; un número que comienza con un cero se considera que está en octal.

En c la selección se maneja mediante la clase usual de sentencia IF, que toma la forma:

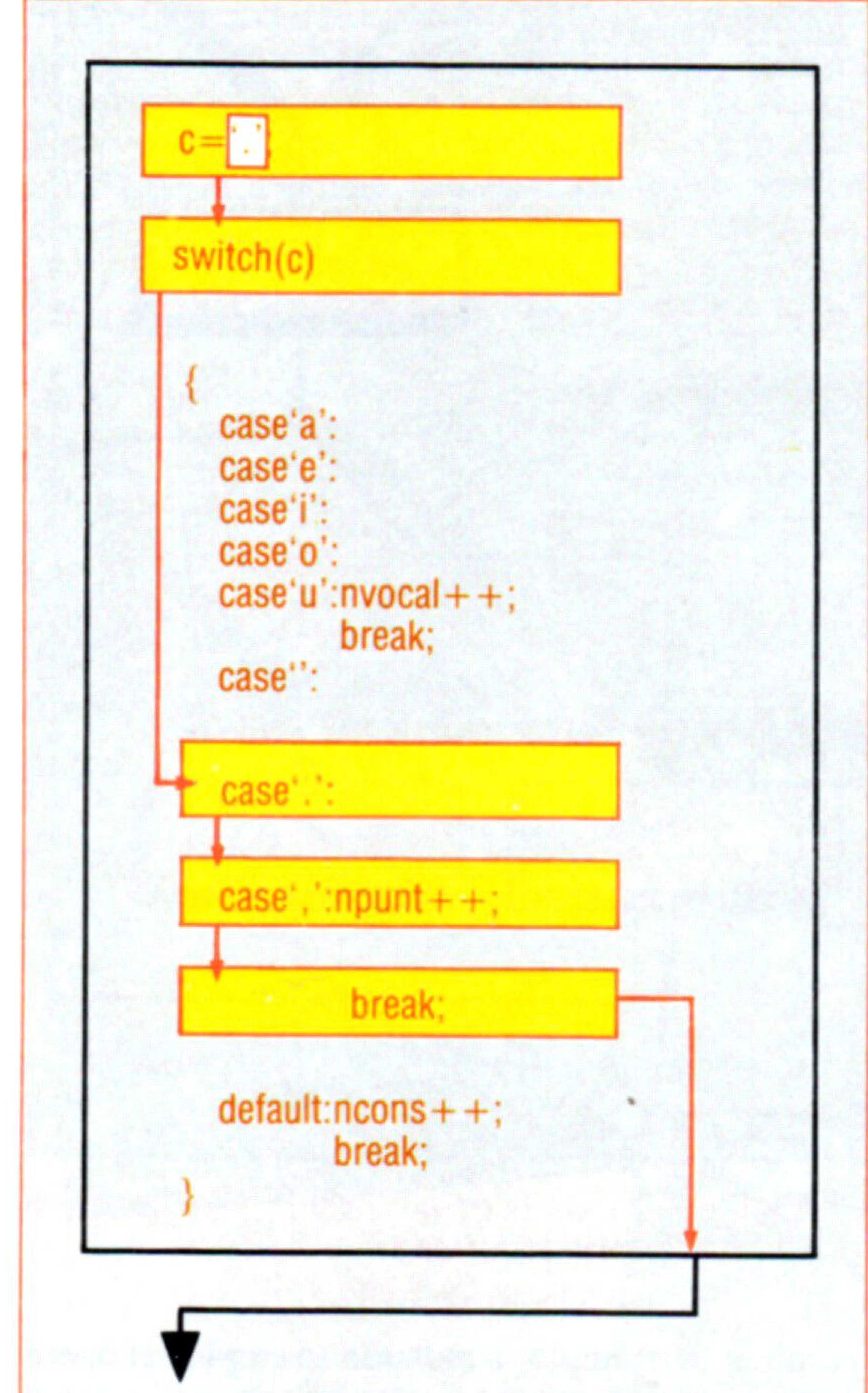
```
if(expresión)
  sentencia_1
else
  sentencia_2
```

donde la expresión normalmente es una expresión lógica, pero, como ya hemos visto, también puede ser cualquier expresión que dé un valor entero. Observe que no hay ningún then y, como siempre, la parte else es opcional.

Las sentencias pueden ser simples o compuestas encerradas entre llaves {}. Los ifs anidados pueden ser un problema, tal como sucede en otros lenguajes, pero se aplica la regla general, de modo que cada else se empareja con el if más reciente. Si usted desea cambiar esta situación, puede utilizar llaves.

Las selecciones más complejas, a partir de más de dos alternativas, se pueden tratar mediante la construcción switch, que desempeña la misma función que la sentencia case del PASCAL. El formato de switch es:

```
switch (expresión_entera)
{
case valor_1:sentencia_1;
case valor_2:sentencia_2;
.........
default:sentencia_defecto;
}
```



Control de "switch"

La construcción SWITCH actúa de modo muy similar a una construcción ON...GOTO de BASIC. Se evalúa el parámetro de SWITCH y se pasa el control al CASE adecuado. El ejemplo que vemos suma 1 a la variable contadora adecuada (nvocal, npunt o ncons) según el valor de la variable c. Esto proporciona un medio para contar cuántas veces se producen vocales, consonantes y signos de puntuación. Normalmente la variable c sería un valor de entrada, pero aquí, para mostrar el flujo de control en un caso determinado, se le ha asignado el valor '.'. Observe que la "bajada" desde un CASE al siguiente proporciona una forma conveniente de dar cabida a múltiples casos con la misma acción. El "break" final no es estrictamente necesario, pero constituye una buena práctica en caso de que en el futuro usted decida incluir un CASE extra al final

Los valores de case deben ser constantes y todos diferentes; la sentencia por defecto es opcional. Los valores de case sirven como etiquetas de modo que la ejecución no se transfiera automáticamente al final de la sentencia de case cuando ya se haya cumplido uno de los casos. En cambio, "baja" hasta la siguiente etiqueta. Para impedir esto se puede utilizar la sentencia break para transferir el control a la sentencia que sigue tras el switch. Veamos un ejemplo:

La iteración, o *looping*, se puede tratar de tres formas.

La construcción más utilizada es el bucle while, que asume la forma:

```
while(condición)sentencia;
```

Como es habitual, la sentencia puede ser simple o compuesta (entre llaves).

También es común, aunque no constituye una buena práctica, colocar la mayor parte del cuerpo del bucle en la condición, recordando que la condición puede ser cualquier expresión de enteros. Los siguientes ejemplos representan dos maneras de escribir el mismo bucle para entrar una secuencia de dígitos como caracteres y convertirlos en un número entero, terminando con el primer no dígito:

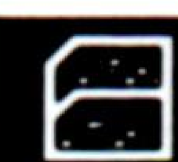
```
int num;
char c;
num=0;
c=getchar();
while(c)='0'&&c(='9')
{
  num=num*10+c-'0';
  c=getchar();
}
```

Observe el modo en que se pueden utilizar los caracteres c y '0' en aritmética. En estos casos se toma automáticamente el código ASCII y los valores se convierten en el tipo int para el cálculo. El segundo procedimiento coloca a la sentencia c en la condición del bucle:

```
num=0;
while((c=getchar())>='0'&& c<='9')
  num=num*10+c-'0';
```

Observe aquí que sólo hay una única sentencia básica a repetir, de modo que no hay necesidad de utilizar las llaves.

La segunda forma de bucle del C es utilizar la sentencia for. Ésta es similar, en algunos sentidos, al bucle FOR...NEXT del BASIC, en cuanto a que hay una inicialización, una condición terminadora (o continuadora) y una "función de paso" que se repite a cada pasada del bucle. En BASIC, como en la mayoría de los otros lenguajes, la inicialización es un valor entero en una variable entera; la condición terminadora es un valor final que debe alcanzar esta variable, y la función de paso tan sólo añade un valor entero a la variable. En C, esta sentencia es mucho más general:



```
for(exp1;exp2;exp3)
   sentencia;
```

donde exp1 representa la inicialización y se lleva a cabo antes de que la sentencia se ejecute por primera vez. La condición de continuación, exp2, y el bucle se repetirán mientras esta expresión permanezca verdadera (o no cero); exp3 se lleva a cabo al final de cada repetición de la sentencia. En BASIC:

```
FOR I=1 TO N
sentencia(s)
NEXT I
```

tiene su equivalente en C con:

```
for(i=1;i<=n;i++)
sentencia;
```

Para ilustrar la flexibilidad de la construcción for y la potencia del lenguaje, podríamos codificar el mismo bucle del ejemplo del bucle WHILE para convertir una serie de caracteres de entrada en un número entero del siguiente modo:

```
for(num=0;(c=getchar())>='0'&& c<='9';
   num=num*10+c-'0');
```

Observe que todo el cuerpo del bucle está incorporado en la condición. Esto suele ser posible en C, pero es discutible si constituye o no un buen hábito. El código producido es, ciertamente, compacto y eficiente, pero también es muy críptico.

En C existe una tercera forma de bucle, que es el equivalente del bucle repeat...until del PASCAL. A diferencia de las otras dos construcciones, la comprobación se realiza al final del bucle en vez de al principio, y toma esta forma:

```
do
   sentencia
while(condición);
```

Esta construcción, sin embargo, se utiliza muy raramente.

"Break" y "continue"

Existen dos sentencias adicionales que hacen que los bucles resulten bastante más fáciles de usar que en otros lenguajes. Ya nos hemos encontrado antes con la sentencia break. Si se encuentra un break en medio de un bucle, la ejecución se transferirá automáticamente a la sentencia que sigue al final del bucle. El siguiente ejemplo leerá una sucesión de caracteres terminando con un Return, pero también saldrá del bucle si se entra un Control C (código 3 ASCII):

```
while((c=getchar())!='\n')
{
   if c==3
      break
   else
         /*seguir procesando c*/;
}
```

La sentencia trabaja de modo similar, pero simplemente pasa a la siguiente ejecución del bucle en vez de interrumpirse por completo. No es utilizada muy comúnmente.

La existencia de break y continue elimina casi totalmente la necesidad de una sentencia goto. En C hay un goto, junto con un mecanismo para etiquetar sentencias, pero nunca es necesario emplearlo, y hacerlo es una práctica desaconsejable. Después de haber dado fe de su existencia, ya no volveremos a mencionarlo. Si realmente usted desea escribir un programa en C utilizando gotos, le bastará cualquier libro de texto sobre este lenguaje.

"El C de un vistazo"
El libro *C at glance* (El C de un vistazo) ofrece un tratamiento del tema bastante más en profundidad de lo que sugiere el título. La obra presupone el conocimiento del BASIC por parte del lector y enfoca el C desde este supuesto, haciéndolo fácilmente accesible a la mayoría de programadores de ordenadores personales. Su autor es Adam Denning y ha sido publicado por la editorial británica Chapman and Hall

"Un libro sobre C"
A book on C (Un libro sobre C) analiza exhaustivamente el lenguaje y explora temas relacionados con el mismo, en particular el Unix, con el cual el C está estrechamente relacionado. Sus autores son Ira Pohl y Al Kelley

Imperativo primordial

```
/* Programa para imprimir todos los números
primos menores que 1000*/
main()
{
   int contador_primos=0,límite=1000,divisor,
   número_a_comprobar;
/* Observe la inicialización de las dos variables,
contador_primos y límite, en la sentencia de
declaración*/
   for(número_a_comprobar=2,número_a_
   comprobar<límite,++núm._a_comprobar)
/* bucle principal para comprobar todos los
números entre 2 y 1000*/
   {
      divisor=1;
/* buscar divisores utilizando el operador%(mod)*/
   while(núm._a_comprobar%++divisor !=0);
/* el incr. se realiza antes de la comprobación*/
/* el número es primo si el divisor alcanza el valor
del número_a_comprobar*/
   if (divisor==número_a_comprobar)
/* sumar 1 al contador*/
   {
      ++contador_primos;
/* imprimir número primo, diez en una línea*/
         printf ("%6d",número_a_comprobar);
         if(contador_primos % 10==0)
            printf("/n");
   }
}
printf("\n\nhay %d números primos menores que
,"n/b0\0
   contador_primos,límite);
}
```

Un ejemplo de lo mejor
Nuestro programa de muestra cuenta e imprime todos los números primos entre 2 y 1000. Observe el compacto código producido utilizando la instrucción de preincrementado ++ y el formato del bucle FOR

Siga las instrucciones

Iniciamos aquí el estudio detallado de las instrucciones del 68000 de Motorola. Empezaremos por las más sencillas

En los dos capítulos anteriores hemos examinado la forma en que el 68000 direcciona sus operandos y hemos mostrado el empleo de algunas instrucciones (tales como MOVE, ADD y la generalización 'OPCODE'). Demos un paso más para estudiar el conjunto de instrucciones con más detalle, comenzando por las instrucciones de copia de datos, para pasar después a las instrucciones de cálculo que conciernen a la aritmética en sistema de numeración binario.

Vamos a demostrar la utilidad de instrucciones más importantes, y a resaltar todo detalle relevante o posible trampa en su empleo. Si usted pretende programar el 68000 con frecuencia, habrá de considerar, sin embargo, la adquisición o bien del *Manual del usuario del 68000 de Motorola* o bien alguna de las restantes publicaciones que mencionamos en este curso.

Antes de estudiar las instrucciones hemos de examinar con algún detalle el contenido del registro de estado (SR: *status register*). La mitad de este registro contiene los códigos de condición que indican el resultado de la última instrucción ejecutada. Cada código de condición puede ser considerado como memoria de un bit asignada a una condición aritmética particular. Examinémoslos uno por uno:

- *BIT 0: bit de arrastre o bit C:* Este bit se pone a uno cuando la operación aritmética produce un arrastre en el bit más significativo del operando de datos. Por ejemplo:

la suma de	0110 0000
y	1110 0000
da	1 0100 0000

En este caso, se arrastra un 1 del bit más significativo de la suma, que tiene un byte de longitud.

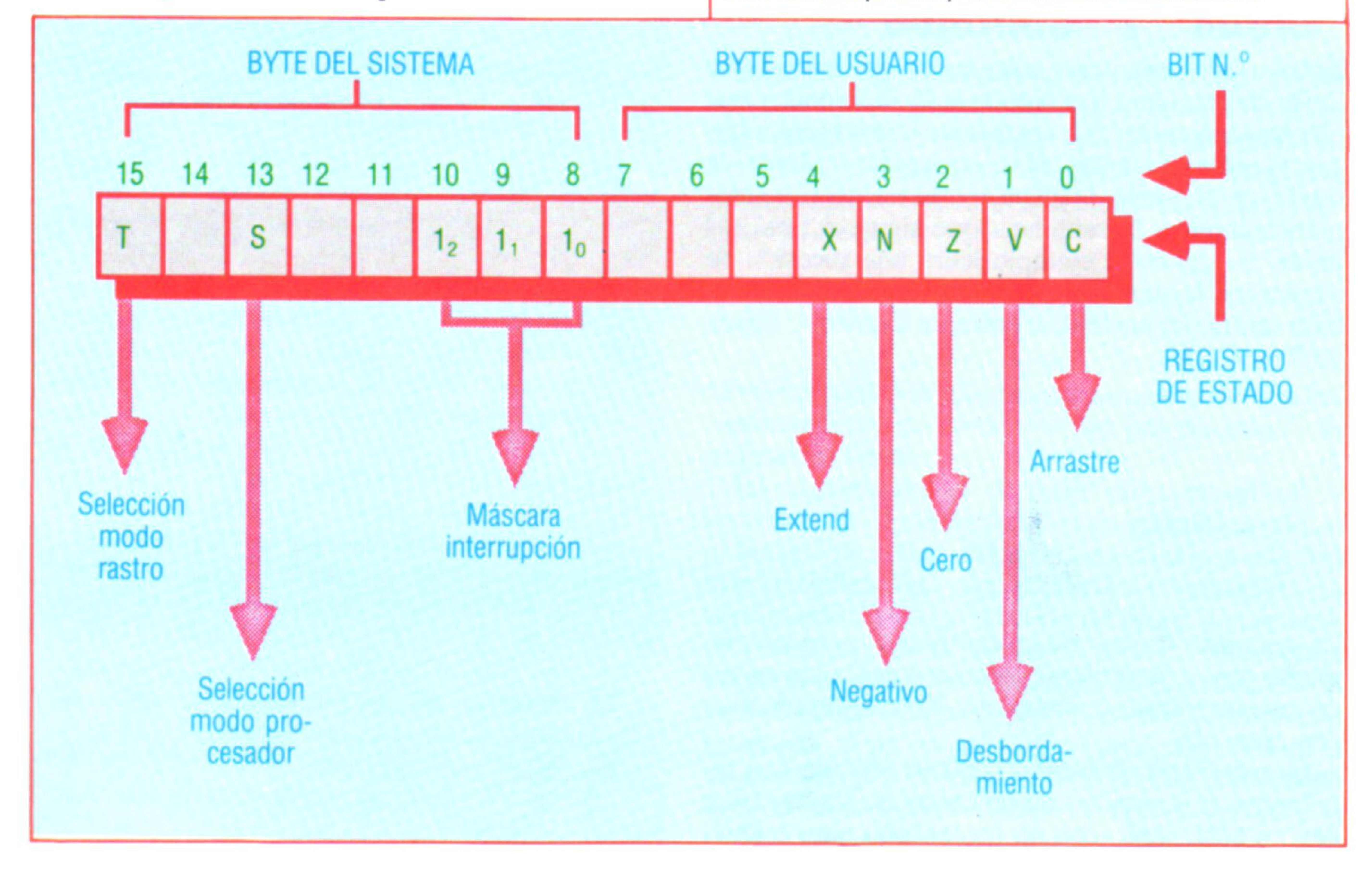
En este ejemplo el arrastre es llevado al bit C del SR, y no al bit menos significativo de la siguiente unidad de datos (en este caso, una palabra). Es de notar también que el arrastre puede ser significativo o no, según lo que se esté haciendo. Si, por ejemplo, se está calculando algún resultado de precisión múltiple (sobre otras unidades del operando de datos), es claro que el bit C debe ser significativo.

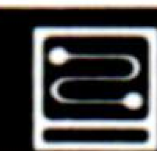
- *BIT 1: bit de desbordamiento o bit V:* Se activa cuando el resultado del cálculo no se ajusta al inter-

Estado mayor

El registro de estado de 32 bits se divide en las secciones del sistema y del usuario, cuyo significado de bits se muestra aquí. Obsérvense los bits de estado empleados para señalar mediante flag los modos del procesador: el 68000 puede operar en dos modos, el supervisor y el usuario. La razón de esto es que en un entorno de multiproceso es necesario ser capaz de controlar las diferentes tareas que se ejecutan. Por lo general, las tareas se ejecutan en el modo usuario (que posee su propio puntero de pila y no permite la alteración de los bits de estado del sistema), y el software de sistema (como sería la supervisión de las tareas) se ejecuta en el modo supervisor.

Para comenzar, es frecuente que usted entienda más conveniente el empleo del modo supervisor del 68000, en el cual son legales todos los opcodes, y desentenderse del problema de evitar las instrucciones privilegiadas en el modo usuario. Además, el 68000 posee un modo *trace* (rastro), que es una facilidad muy eficaz a la hora de depurar errores y que permite al usuario rastrear los pasos de las instrucciones, mientras que el 68000 llama automáticamente a la rutina de depuración para el usuario tras cada instrucción





valo de bits de los operandos de datos. Por ejemplo, si se suma un 1 a 32767 (el número entero positivo máximo para una palabra de 16 bits), se obtiene un desbordamiento en los operandos para datos de palabras, y el resultado binario carecerá de significado.

- *BIT 2: bit cero o bit 2:* Se activa cuando el resultado de un cálculo previo es cero.

- *BIT 3: bit negativo o bit N:* Se activa en los resultados negativos.

- *BIT 4: bit de extensión o bit X:* Se emplea en operaciones de multiprecisión, pero en general equivale al bit C (aunque no viene afectado por las instrucciones MOVE).

La instrucción del 68000 para copiar datos es MOVE, que copia de una fuente a un destino. Es una instrucción versátil: se puede emplear con ella cualquier modo de direccionamiento como fuente y casi la mayoría de los modos de direccionamiento como destino (a excepción de los registros de direcciones, los modos relativos al PC y el modo inmediato). Este grupo de modos de direccionamiento se conoce como *modos de datos alterables,* y existe un subgrupo dentro de éste denominado *modos de memoria alterable* (datos alterables menos registros de datos). De ambos grupos nos ocuparemos más adelante.

Volviendo a cómo se ha de emplear la instrucción MOVE, se advertirá que ninguna de las instrucciones MOVE siguientes es lícita:

```
RORG $1000
MOVE D2,MIMI   el PC relativo a MIMI no es válido
MOVE D2,A2     no es válido el registro de
               direcciones como destino
```

Nótese que la instrucción MOVE afecta sólo a los bits N y Z del SR, y que los bits V y C serán puestos a cero.

Para solventar el problema de los registros de direcciones como operandos de destino, se dispone de dos opciones:

- Emplear MOVEA, que toma el contenido del operando fuente y lo copia en el registro destino de direcciones.
- Emplear LEA, que toma la dirección fuente (por lo general, absoluta) y la copia en el registro de direcciones.

Ninguna de estas dos instrucciones afectará a los códigos de condición. Del mismo modo, hay instrucciones especiales para llevar los datos desde y hacia el registro de estado y el puntero de pila del usuario, pero son instrucciones que tienen que ver sobre todo con la programación de sistemas.

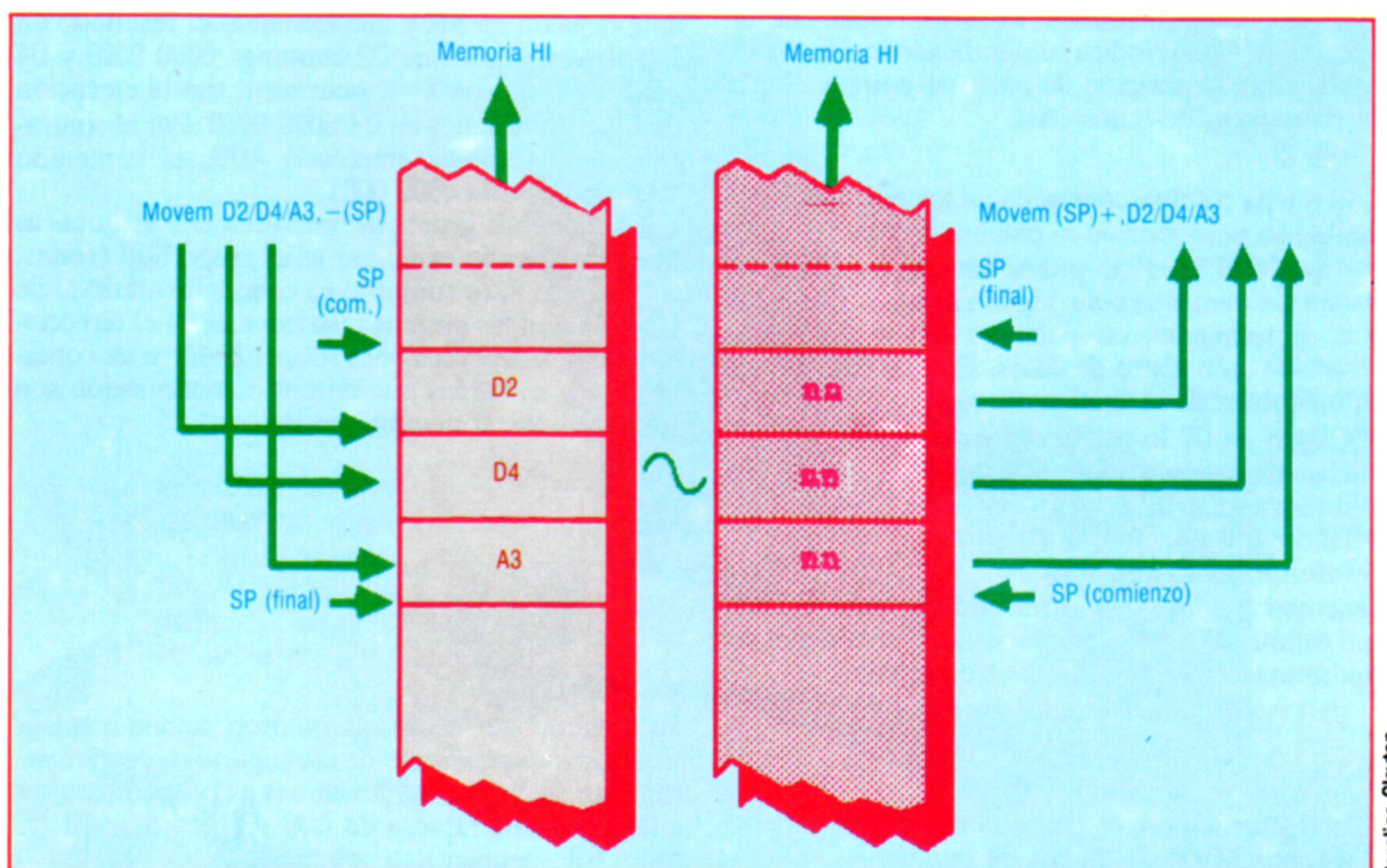
Otra instrucción para copiar datos extraordinariamente poderosa es MOVEM, que nos permite guardar o recuperar cualquier registro declarado (de direcciones o de datos) desde o hacia posiciones de memoria consecutivas. Esto significa que al entrar en una subrutina todos los registros que de otra manera perderían su validez en ella pueden preservarse y a la salida recuperarse. Por ejemplo:

```
entrada MOVE    D2/D4/A3, PAD
                (el código de la subrutina
                emplea D2, D4 y A3)
salida MOVEM    PAD, D2/D4/A3
```

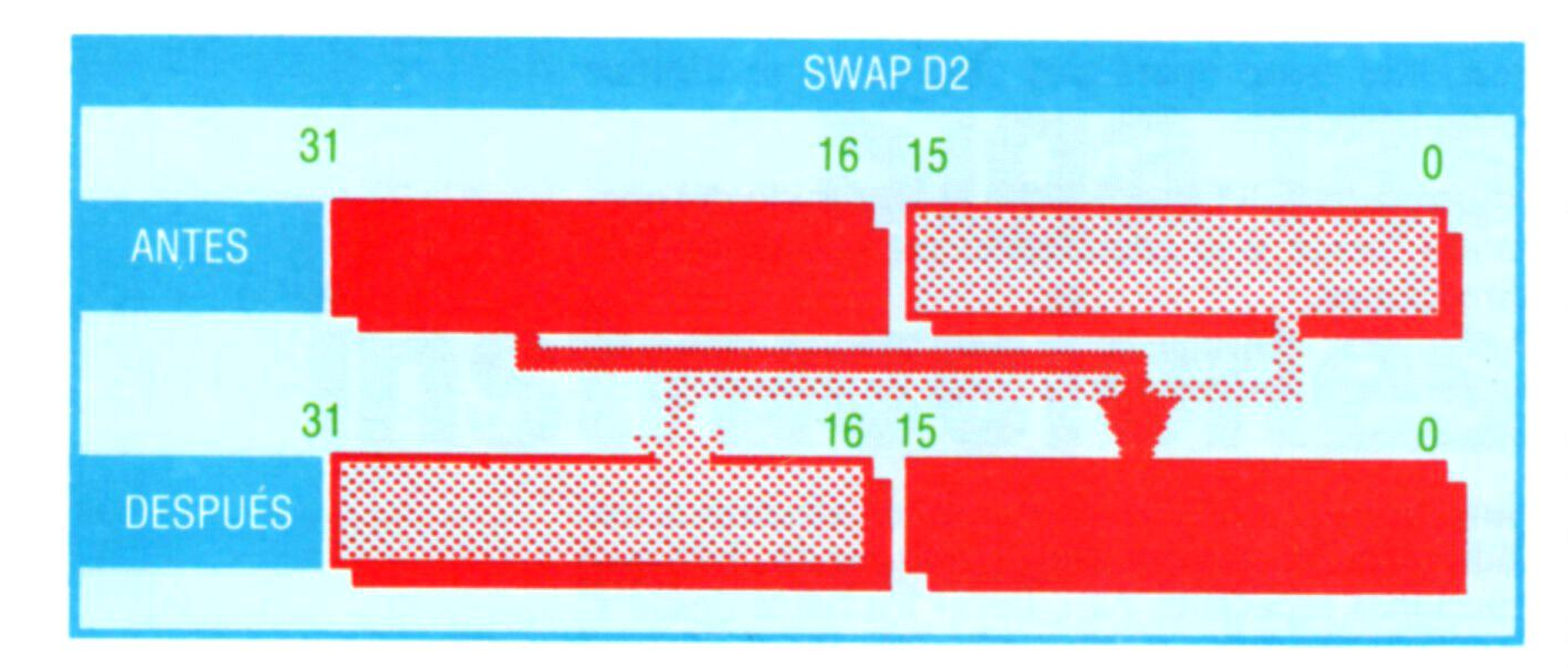
Con ello se guardarán D2, D4 y A3 en PAD a la entrada, y los tres registros se restaurarán a la salida. Los registros se pueden guardar también en la pila. Así, por ejemplo, podríamos tener:

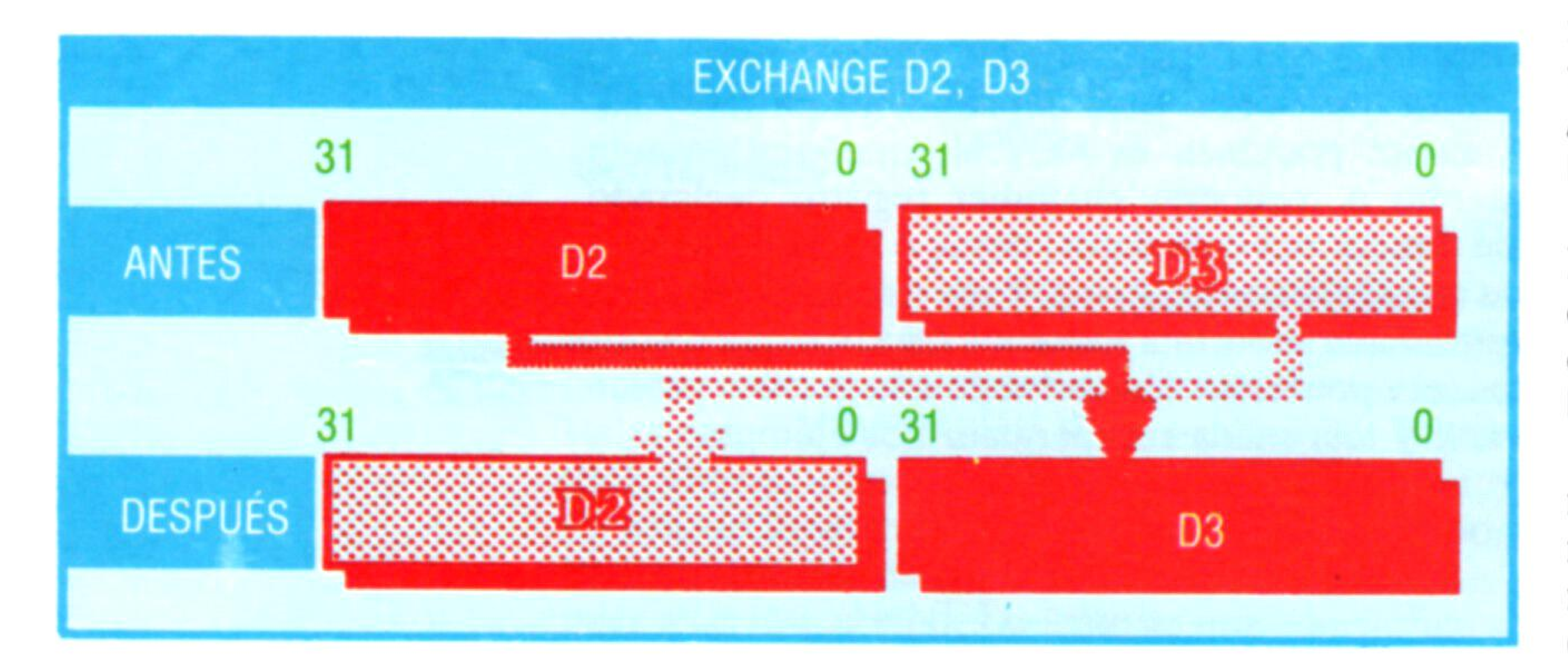
```
entrada MOVEM   D2/D4/A3,-(SP)
                código de la subrutina
salida MOVEM    (SP)+,D2/D4/A3
```

Otra variante de la instrucción MOVE es la instrucción rápida *(quick)* MOVE (o MOVEQ). Es útil cuando se establecen constantes de ocho bits con signo (desde +127 hasta −128) en un registro de datos dentro de una palabra de memoria. Entre los em-



Vayan pasando, por favor
La poderosa instrucción MOVEM nos permite manipular los "listados de registros" en secuencia dentro de una instrucción. Mostramos aquí su uso junto con las instrucciones predecremento y postincremento para poner y hacer salir tres registros hacia y desde la pila. La posición del puntero de la pila es la que se muestra antes y después de cada instrucción





Dos tipos de intercambio
El empleo del almacenamiento de palabras largas puede resultar más bien pesado cuando son necesarios operandos con longitud de un byte o una palabra. SWAP nos ayuda en este problema intercambiando los valores contenidos en los bits del 0 al 15 de un registro con los contenidos en los bits del 16 al 32. Además, EXG (la instrucción de intercambio) permite el intercambio de palabras largas, como se muestra en el esquema inferior

pleos más comunes estará el establecer contadores de bucles dentro de un registro de datos. Por ejemplo:

```
MOVEQ #34,D2
```

establecería 34 en D2 en una palabra de memoria. Nótese que si se quita la Q algunos ensambladores no asumen el modo rápido. Por esto, la instrucción MOVE #34,D2 se codificaría en dos palabras.

Otra instrucción para copiar datos de posible utilidad para operaciones de la pila es PEA *(push effective address:* poner la dirección efectiva... en la pila). Por ejemplo, PEA BETTY llevará la dirección de BETTY a la pila: hará –(SP).

Examinemos finalmente las instrucciones de intercambio. Como indica su significado inglés, SWAP intercambia la posición de palabras enteras dentro de un registro de datos. Así:

```
SWAP D2
```

hará que la palabra contenida en los bits del 0 al 15 cambie su posición con la palabra contenida en los bits del 16 al 31. Esto puede ser útil si por un supuesto deseamos repetir algún cálculo sobre palabras almacenadas en palabras largas completas dentro de un registro de datos. En este caso el procedimiento sería el siguiente:

- Cargar en D2 la palabra larga completa
- Hacer el cálculo sobre la palabra
- Intercambiar D2
- Hacer cálculo sobre la palabra
- Intercambiar la palabra

Tenemos también la instrucción de cambio EXG, que cambia palabras completas de 32 bits según un determinado modelo. Algunos ejemplos:

```
EXG D2,D3   Intercambio entre registros de datos
EXG A3,A4   Intercambio entre registros de dirs.
EXG D2,A5   Intercambio de datos y direcciones
```

Esta instrucción no es más que una SWAP para palabras completas de 32 bits de tamaño.

Aritmética con enteros

Estas instrucciones forman la base de todos los cálculos aritméticos (ya sean con fracciones, valores reales o de doble precisión) que, básicamente, consistan en sumar números binarios. Aun cuando la aplicación que usted desea no implique cálculo numérico alguno, necesita saber cómo realizar las operaciones aritméticas más sencillas a fin de que, por ejemplo, pueda convertir códigos de caracteres o formar índices de tablas.

La instrucción ADD se limita a sumar la fuente con el destino y almacenar el resultado en el destino. Los objetos de datos pueden ser de cualesquiera tamaños de atributos de datos y son afectados todos los códigos de condición. Por ejemplo, ADD.W D2,D3 sumará el contenido de palabra de D2 con D3 y almacenará el resultado en D3.

Para los datos fuente se pueden emplear todos los modos de direccionamiento, pero es necesario el empleo de una instrucción especial, ADDA, cuando el destino es un registro de direcciones. Así, ADDA D2,A4 sumará el contenido de D2 en A4.

Para el caso de datos inmediatos, está también la instrucción ADDI. Esta instrucción suma el dato inmediato (se permiten todos los atributos) almacenado como una palabra de extensión en el destino (se permiten sólo los modos alterables de datos). Así, ADDI #3423, BETTY sumará 3423 al contenido de BETTY.

A semejanza de MOVE, también para ADDI disponemos de una forma rápida, ADDQ. Pero el empleo de esta última instrucción sólo admite datos dentro del intervalo del 1 al 8. Así, ADDQ 5,D2 sumará 5 al contenido de D2, y la instrucción entera ocupa una palabra.

Hay que hacer una puntualización importante sobre las instrucciones ADD que hemos examinado hasta aquí: y es que no incluyen arrastre alguno en la suma destino. Si queremos esto (en especial para operaciones aritméticas en doble precisión) deberemos emplear la instrucción ADDX. Así ADDX D2,D4 sumará los contenidos de D2 y D4, poniendo el bit de extensión en SR y almacenando el resultado en D4. Supongamos que D2 contenga 0000 0000 y D4 sea 0000 0001 con X=1; pues bien, tras la ejecución de ADDX tendremos en D4 0000 0010. Por el contrario, si hubiéramos empleado ADD, el contenido final de D4 sería 0000 0001.

El siguiente grupo de instrucciones aritméticas que es necesario examinar es el grupo SUB (restar, o *subtract*). Este conjunto es complementario al de las ADD, con las mismas restricciones en el direccionamiento y repercusiones sobre el código de condición. Los ejemplos que damos a continuación son todos válidos al tiempo que típicos:

```
SUB  D2,D3     Restar D2 de D3
               y poner el resultado en D3
SUBA #4,A3     Restar 4 de A3
SUBI #200,D2   Restar 2 de D2
SUBQ #1,D2     Resta rápida D2 menos 1
SUBX D2,D4     Da a D4 el resultado D4-D2-X
```

Es de notar que Motorola no proporciona una instrucción independiente de incremento o decremento. Para realizar estas funciones es preciso recurrir a las versiones rápidas de ADD y SUB (¡después de todo, sólo ocupan una palabra!).


GRAN NOVEDAD EDITORIAL
PARA TI, PROFESIONAL EN ACTIVO.....
PARA TI, FUTURA SECRETARIA.....
enciclopedia de la
SECRETARIA
de
PLANETA-AGOSTINI
PIDELA EN TU QUIOSCO
o suscríbete ahora llamando al (91) 415 97 12
... para estar al día
... para ser más eficaz
... para actualizar tus estudios
... para encontrar mejor empleo

10112

9 788485 822836